September 1979
vol 5, no 9
$2.00

creative computing

the #1 magazine of computer applications and software

Educational Computing:
- 12 Articles
- 6 Applications that you can use

In-depth Evaluations:
- Compucolor II
- Classmate 88
- TI 810 Line Printer
- CBASIC
- Heath H17 Disk System

Plans to build a cassette tape monitor

Computer Trivia Contest — You can be a winner!

Columns: TRS-80, PET, Q&A, Software Legal Forum

Computer Chess — The Levy Wager

Can Computers Think? (Part 2)

Cover Story: A Dozen Apples for the Classroom. Page 52

"31,000 student hours later, we still love Apple Computer."

Dr. Kennetl
Brumbaugl
Minnesota
Educationa
Computing
Consortiun

When the Minnesota Educational Computing Consortium recommended Apple Computer to the state's school districts—well, it started something big.

Today there are hundreds of Apple Computers in use in 35% of Minnesota's elementary and secondary schools, and nearly all of the colleges and universities in the state. Most communicate with the Consortium's CYBER 73 mainframe in a state-wide educational computer network.

The educational computer

Dr. Kenneth Brumbaugh, Manager of User Services, heads the team responsible for supporting instructional computing.

"MECC evaluated personal computers and chose Apple because it was the one that met our rather rigid specifications.

"And, we employ a conventional timesharing system, with remote terminals. But that means high phone costs. And limited user access. Apple solves that. It gives schools a stand-alone computer for about the price of a terminal. Also, Apple interfaces directly to our CYBER, so we can download programs to any Apple in the state. That also means we can serve as the communication link for the wealth of new programs students and teachers are writing themselves. For us, Apple is an excellent educational computer.

The kids—and the teachers—love Apple

"One big reason we chose Apple is that it is so easy to program. Now, with Pascal, Apple can provide even more programming flexibility.

"For example, MECC has written a note-recognition program to help teach music that takes advantage of Apple's unique built-in speaker. And Apple's color graphics make programs far more interesting than conventional black and white terminals can.

"To date, we've logged over 31,000 student hours on Apple Computers. We even have schools trying out computers for home study. The kids love the Apple. And so do the teachers."

Is Apple for you?

For the name and address of your local Apple dealer and your free copy of Apple's *new* Curriculum Materials Kit, call **800-538-9696**. In California, **800-662-9238**. Or write us at 10260 Bandley Drive, Cupertino, CA 95014.

CIRCLE 108 ON READER SERVICE CARD

apple® computer

Low-cost hard disk computers are here

11 megabytes of hard disk and 64 kilobytes of fast RAM in a Z80A computer for under $10K. Two floppy drives, too. Naturally, it's from Cromemco.

It's a reality. In Cromemco's new Model Z-2H you get all of the above and even more. With Cromemco you get it all.

In this new Model Z-2H you get not only a large-storage Winchester hard disk drive but also two floppy disk drives. In the hard disk drive you get unprecedented storage capacity at this price—11 megabytes unformatted.

You get speed—both in the 4 MHz Z80A microprocessor and in the fast 64K RAM which has a chip access time of only 150 nanoseconds. You get speed in the computer minimum instruction execution time of 1 microsecond. You get speed in the hard disk transfer rate of 5.6 megabits/sec.

EXPANDABILITY

You get expandability, too. The high-speed RAM can be expanded to 512 kilobytes if you wish.

And the computer has a full 12-slot card cage you can use for additional RAM and interface cards.

BROADEST SOFTWARE SUPPORT

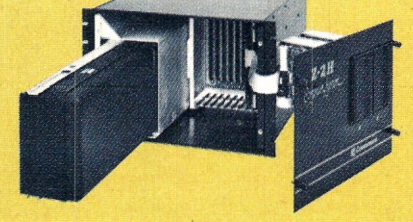
With the Z-2H you also get the broadest software support in the microcomputer field. Software Cromemco is known for. Software like this:

- **Extended BASIC**
- **FORTRAN IV**
- **RATFOR (RATional FORtran)**
- **COBOL**
- **Z80 Macro Assembler**
- **Word Processing System**
- **Data Base Management**

with more coming all the time.

SMALL, RUGGED, RELIABLE

With all its features the new Z-2H, including its hard disk drive, is still housed in just one small cabinet.



Hard disk drive at lower left can be interchanged just by sliding out and disconnecting plug. Seven free card slots are available. Z-2H includes printer interface card.

Included in that cabinet, too, is Cromemco ruggedness and reliability. Cromemco is time-proved. Our equipment is a survey winner for reliability. Of course, there's Cromemco's all-metal cabinet. Rugged, solid. And, there's the heavy-duty power supply (30A @ 8V, 15A @ +18 V, and 15A @ −18V) for circuitry you'll sooner or later want to plug into those free card slots.

CALL NOW

With its high performance and low price you KNOW this new Z-2H is going to be a smash. Look into it right now. Contact your Cromemco computer store and get our sales literature. Find out when you can see it. Many dealers will be showing the Z-2H soon—and you'll want to be there when they do.

PRESENT CROMEMCO USERS
We've kept you in mind, too. Ask about the new Model HDD Disk Drive which can combine with your present Cromemco computer to give you up to 22 megabytes of disk storage.

Cromemco
incorporated
280 BERNARDO AVE., MOUNTAIN VIEW, CA 94040 • (415) 964-7400
Tomorrow's computers now

CIRCLE 135 ON READER SERVICE CARD

Tinker, Tailor, Soldier, Sailor . . . Doctor, Lawyer . . . the Chieftain's here.

No matter whether you're a serious hobbyist or a serious businessman, the Chieftain 6800 microcomputer with capabilities that surpass the Z-80 is made for you.

Smoke Signal's quality-packed Chieftain I features two 5.25-inch mini-floppy drives and Chieftain II features two 8-inch floppy drives.

Both microcomputers provide 32K static memory, two serial I/O ports, a 2 MHz processor board, a 2K RAM monitor, a nine-slot motherboard with built-in baud rate generator and gold connectors for high reliability. The Chieftain's stylish leather-grained cabinet houses the above with its own cooling fan and regulated power supply.

Every Chieftain is complete with system software and is totally burned-in as well as tested to further insure high reliability.

And it's expandable to 64K memory with up to 2 megabytes floppy disk storage.

So see your nearest Smoke Signal dealer, he'll be glad to show you how to get your wampum's worth. Systems start at $2,595.

☐ Send information on your Chieftain microcomputer
☐ Send name of nearest dealer

Name____________________

Address____________________

Company ____________________

City ____________________

State/Zip ____________________

SMOKE SIGNAL BROADCASTING

31336 Via Colinas, Westlake Village, California 91361, (213) 889-9340

Dealer inquiries invited.

Hail to the Chieftain

Smoke Signal Broadcasting, 31336 Via Colinas, Westlake Village, CA 91361, (213) 889-9340

In This Issue

articles

evaluations & profiles

applications ~ games

fiction & foolishness

departments

September 1979
Volume 5, Number 9

Creative Computing magazine is published monthly by Creative Computing, P.O. Box789-M, Morristown, NJ 07960. (Editorial office: 51 Dumont Place, Morristown, NJ 07960 Phone: (201) 540-0445.)

Domestic Subscriptions: 12 issues, $15, 24 issues $28, 36 issues $40. Send subscription orders or change of address (P.O. Form 3575) to Creative computing, P.O. Box 789-M, Morristown, NJ 07960. Call 800-631-8112 toll-free (in New Jersey call 201-540-0445) to order a subscription (to be charged only to a bank card).

Second class postage paid at Morristown, New Jersey and at additional mailing offices.

OK to Reprint

Material in *Creative Computing* **may be reprinted without permission** by school and college publications, personal computing club newsletters, company house organs, and non-profit publications. Only original material may be reprinted; that is, you may not reprint a reprint. Also, each reprint must carry the following notice on the first page of the reprint in 7-point or larger type (you may cut out and use this notice if you wish):

Copyright © 1979 by Creative Computing
51 Dumont Place, Morristown, NJ 07960
Sample issue $2.00;
One-year subscription $15.00

Please send us two copies of any publication that carries reprinted material. Send to attention: David Ahl.

Advertising Sales

Advertising Coordinator
Marcia Wood
Creative Computing
93 Washington Street
Morristown, N.J. 07960
(201) 540-9168

Western States, Texas
Jules E. Thompson, Inc.
1290 Howard Ave., Suite 303
Burlingame, CA 94010
(415) 348-8222

Southern California
Jules E. Thompson, Inc.
2560 Via Tejon
Palos Verdes Estates, CA 90247
(213) 378-8361

Mid-Atlantic, Northeast
CEL Associates, Inc.
36 Sohier Street
Cohasset, MA 02025
(617) 383-6136

New York Metropolitan Area
Nelson & Miller Associates, Inc.
342 Madison Ave., Room 830
New York, N.Y. 10017
(212) 661-9234

Southeast
Warren Langer Associates, Inc.
234 County Line Road
Gilbertsville, PA 19525
(215) 367-0820

4-Year Index

A four-year cumulative index to **Creative Computing** and **ROM** is available. Articles are cross-referenced to both individual issues and collected volumes (**The Best of Creative Computing, Vols. 1 and 2**). Articles are classified by subject area and listed by title and author. Over 2000 items are included. $1.00 postpaid in U.S., $2.00 foreign. Creative Computing, P.O. Box 789-M, Morristown, N.J. 07960.

Back Issues

Back issues of Creative Computing are usually in stock for the current and previous volume. Prices on 1977 (Vol. 3) issues are $2.00 each postpaid or three for $5.00. 1978 (Vol. 4) and 1979 (vol. 5) issues are $2.50 each postpaid, three for $7.00, or six for $12.00

Volumes 1 and 2 are available in book form, The Best of Creative Computing, Vols. 1 and 2. Each book is $10.00 postpaid. Creative Computing, P.O. Box 789-M, Morristown, N.J. 07960

Nine back issues of ROM are available in a complete set for $14.00 postpaid ($17.00 foreign) from Creative Computing.

Microform

Creative Computing is available on permanent record microfilm. For complete information contact University Microfilms International, Dept. F.A., 300 North Zeeb Road, Ann Arbor, MI 48106 or 18 Bedford Road, London WC1R 4EJ, England.

Foreign Customers

Foreign subscribers in countries listed below may elect to subscribe with our local agents using local currency. Of course, subscriptions may also be entered directly to Creative Computing (USA) in U.S. dollars. (bank draft or American Express card). All foreign subscriptions must be prepaid.

Many foreign agents stock Creative Computing magazines, books, and software. However, please inquire directly to the agent before placing an order. Again, all Creative Computing products may be ordered direct from the USA — be sure to allow for foreign shipping and handling.

CANADA	**Surface**	**Air**
1-year	$19	n/a
2-year	36	
3-year	52	

RS-232
Attn: Liz Janik
186 Queen St. W.
Toronto, Ontario M5V 1Z1, Canada

ENGLAND	£	£
1-year	13	22
2-year	25	43
3-year	36	63

CREATIVE COMPUTING
Attn: Hazel Gordon
27 Andrew Close
Stoke Golding, Nuneaton CV13 6EL
England

FRANCE	F	F
1-year	105	180
2-year	200	345
3-year	290	510

SYBEX EUROPE
14/18 Rue Planchat
75020 Paris, France

SWEDEN	kr	kr
1-year	105	180
2-year	200	345
3-year	290	510

HOBBY DATA
Attn: Jan Nilsson
Fack
S-200 12 Malmo 2, Sweden

GERMANY	dm	dm
1-year	46	78
2-year	88	152
3-year	128	224

HOFACKER-VERLAG
Ing. W. Hofacker
8 Munchen 75
Postfach 437, West Germany

HOLLAND, BELGUIM		g1
1-year		90
2-year		175
3-year		250

2 x F IMPORT VAN BOEKEN EN TIJDSCHRIFTEN
Attn: M.F. deVreeze
Postbus 70198
1007 KD Amsterdam, Holland

AUSTRALIA	$A	$A
1-year	23	47
2-year	44	92
3-year	64	136

ELECTRONIC CONCEPTS PTY. LTD.
Attn: Rudi Hoess
Ground Floor, 55 Clarence St.
Sydney, NSW 2000, Australia

JAPAN	Y	Y
1-year	4800	9400
2-year	9200	18,400
3-year	13,500	27,200

ASCII PUBLISHING
305 HI TORIO
5-6-4- Minami Aoyama, Minato-ku
Tokyo 107, Japan

HONG KONG	$HK	$HK
1-year	115	235
2-year	220	460
3-year	320	680

COMPUTER PUBLICATIONS, LTD.
22 Wyndham St., 7th Floor
Hong Kong

PHILIPPINES	P	P
1-year	175	355
2-year	330	690
3-year	480	1,020

INTEGRATED COMPUTER SYSTEMS, INC.
Suite 118, Limketkai Bldg., Ortigas Ave.
Greenhills P.O. Box 483, San Juan
Metro Manila 3113, Philippines

KUWAIT	dinar	dinar
1-year	6.5	11.0
2-year	12.5	21.5
3-year	18.0	31.5

ASCON
Attn: Salah Zeineddin
P.O. Box 4052
Safat, Kuwait

OTHER COUNTRIES		
1-year	$23	$39
2-year	44	76
3-year	64	112

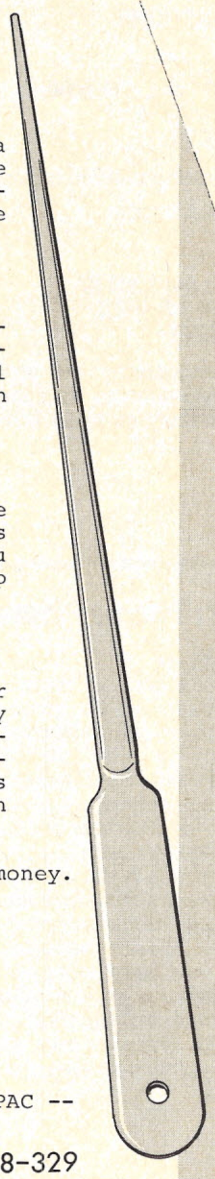
CREATIVE COMPUTING
P.O. Box 789-M
Morristown, N.J. 07960, USA

An Open Letter to Anyone
Selling or Building a System
Around a SOL, Apple or TRS-80 Computer

Exidy inc. Computer Systems Division

Exidy is moving to greatly expanded production facilities. Right now is the time for you to move up to an improved computer system: one built around our Sorcerer computer rather than a SOL, Apple or TRS-80.

The new 48,000 square foot plant (more than double our present size) has the capability of building and testing 200 computers per shift. With more than a year of production under our belts, we are shipping a reliable, well-debugged product.



Compare the Sorcerer to the others:

SORCERER vs SOL

The Sorcerer easily replaces the SOL and provides many extra features for about 40% less money. Sorcerer reads cassette tapes in CUTS format. Most SOL software runs on the Sorcerer with little or no change. And Sorcerer comes with the same I/O ports as SOL. For many extra benefits, read on.

SORCERER vs APPLE

You can ask the Sorcerer to "come out and play." But otherwise it's all business. Standard with Sorcerer are: Floating point BASIC, lower case characters, serial and parallel I/O ports, a numeric key pad, two cassette interfaces with motor control, plus all the features described below.

SORCERER vs TRS-80

On the surface, the TRS-80 may cost less, but compare the capabilities -- Sorcerer is the bargain. With Sorcerer's plug-in ROM PAC software, you're not locked into BASIC. You can even create turn-key application packages that come up running.

SORCERER vs EVERYBODY

Our video display is the winner. Full 128 ASCII character set (including upper and lower case); 30 x 64 screen display (1920 characters on screen); 64 pre-defined graphics characters; 128 user-definable characters; 512 x 240 point resolution. Sorcerer easily expands to a disk system that runs CP/M software; add S-100 bus products with our expansion interface.

All in all, Sorcerer will give your system more value for less money. For more proof, just drop us a note describing your needs.

Sincerely,

Exidy, Inc.

Exidy Incorporated
Computer Systems Division

P.S. This letter was prepared with our new Word Processor ROM PAC -- available now at $99.00 suggested retail price.

390 Java Drive, Sunnyvale, California 94086 (408) 734-9410 Telex 348-329

Input/ Output

Helpful Hint For BASIC

Dear Editor:

At one time or another most BASIC programs will have to initial a large string to spaces (or some other character). The usual method is as follows:

```
10 DIM A$(1000)
•
110 A$ = ""
120 FOR I = 1 TO 1000
130 A$ = A$ + " "
140 NEXT I
```

The following method is ten times faster;

```
10  DIM A$(1000)
•
110 A$ = ""
120 FOR I = 1 TO 10
130 A$ = A$ + A$
140 NEXT I
```

By concatenating the string with itself, the number of characters doubles with each iteration. Only ten repetitions are necessary to yield 1,024 characters. (Two to the tenth power is 1,024.)

Reginald Gates
4244 Carfax
Lakewood, CA

Watch Those Phone Taps!

Dear Editor:

There's nothing more disappointing in an old western movie then when the posse decides to take the law into their own hands to handle the bad guys. It sure sounds like "Colonel Winthrop" in John Craig's **Twice Burned** (July-79) is a villian-to be sure-but when Craig lauds his buddy Brian Allen and his phone-tape system... you should watch your step.

Who said it was legal to tape someone without their knowledge? Better check the books, and one book to check is the telephone book. Ma Bell states - in her general information-that such practices are against company tariffs and not permitted. Eavesdropping and electronic surveillance laws also prohibit such unauthorized acts.

Let's not have your readers misled on this issue - or somebody's liable to get in trouble.

No hard feelings though. Your magazine is great and the article in question was needed. The day I find my issue waiting on my doorstep is always a better day.

James J. Conroy
57 East Garrison Street
Bethlehem, PA 18018

Sometimes it pays to **check a little further**. *Brian Allen's company sells the phone recording device he was using. Therefore, I considered him to be a reliable source for that information. Furthermore, one of the investigators in Tucson, with the Pima County Attorney's office, verified the statement. It does appear, however, that it ain't the proper thing to be doin'.* —*JC*

WOULD YOU LIKE TO EXPRESS YOURSELF IN A PERSONAL VISIT TO AN EDITOR?

In Response To The Mystery Shopper

Dear Editor:

I read the article on the mystery shopper in the June'79 issue of Creative Computing with much interest. Having been on both sides of the counter, I can appreciate Ms. Rust's comments about computer salesmen. Too often, computer stores tend to treat the computer business not as a business but as a game or an ego trip.

I would like to make a few comments, however, regarding the difficulties salesmen have in this business. We at Data Domain have been experimenting with radio and newspaper advertisements, and that has caused some growing pains. We, like most of the stores who got into the business early, had been very informal and treated the whole enterprise as if the store were a meeting place for computer cultists. In addition, due to the lack of stores here in the Midwest, we had done the bulk of our sales over the phone. Now, however, we're getting curious walk-in customers. Suddenly our sales staff (who are mostly hackers themselves) have to be able to assess quickly what type of customer we're dealing with, as well as that person's knowledge of computers. Because we're in a town with a large university, we have to be ready to talk on the "computer science level," as well as on a level of computer knowledge no deeper than that gained by watching Radio Shack advertisements. I hope the customers bear with us. The stereo business, probably the most like ours, does not have to contend with the super-knowledgable as well as with the novice. To give an example of what we're dealing with, our customers range from US Defense Department digital design personnel, to educators who want an easier way to prepare articles, to businessmen who want a more organized way of accounting in their business, to the curious onlooker who is intrigued by all those moving colors on an Apple Star-trek game. Coupled with the problem that no one person can possibly keep up with the entire business, and I think you can see our problem.

Again, thanks for the article. I would hope that you continue to investigate the relationship between computer sellers and buyers.

Data Domain, Inc.
221 W. Dodds St.
Bloomington, IN 47401

NEC introduces The College Board.

Our educational TK-80A–the first complete 8080A based single board computer.

Here's the perfect system for all levels of computer education—from basic computing to advanced programming techniques.

It's a complete 8080A based computer on a single board. With a 25-key pad, 8-digit display, 1-8K byte EEPROM monitor, 1-4K byte RAM, and three 8-bit programmable I/O ports.

And it's fully expandable. Memory can be increased off-board to a total of 64K bytes. And a standard Kansas City interface lets you hook up a cassette for additional storage. If you need a terminal, a TTY or RS 232 interface can be easily attached.

What's more, 2 or 3 TK-80A boards can be connected for instruction in sophisticated programming techniques—such as distributed processing, parallel processing, and peripheral control.

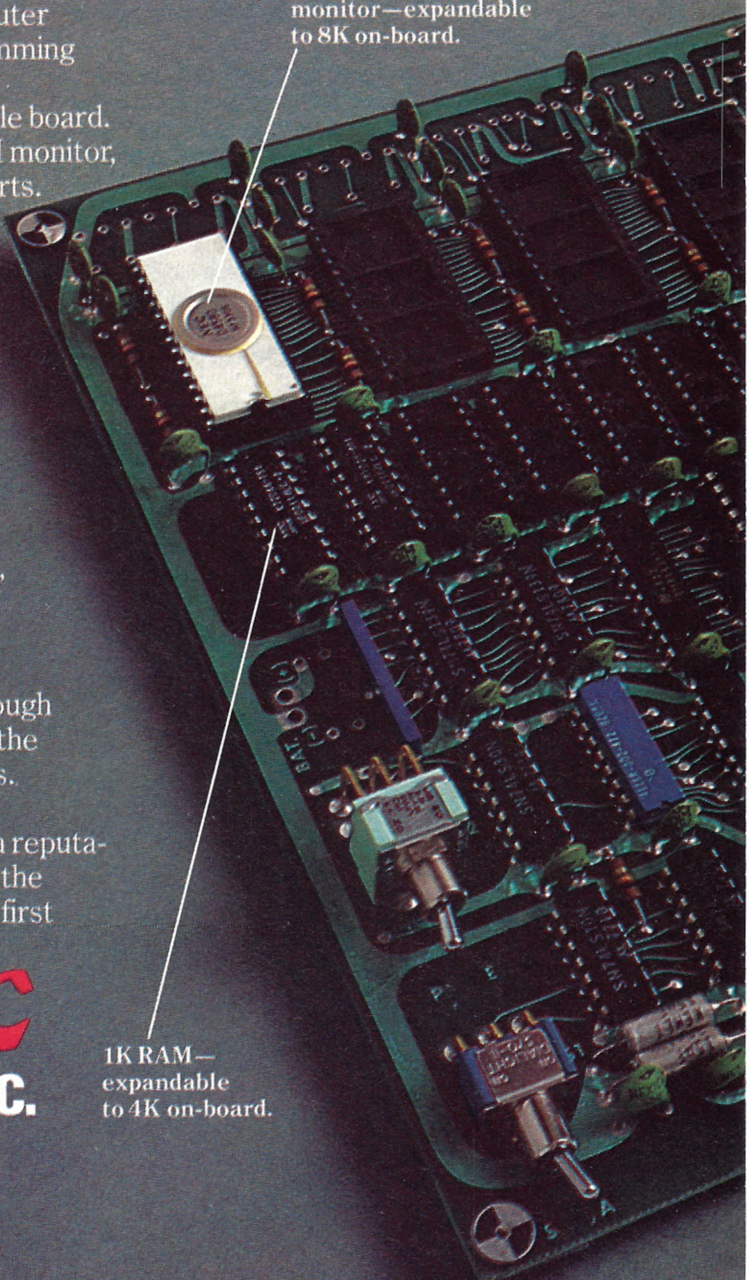
And once students have mastered the TK-80A, they can easily apply what they've learned to process control, energy control systems, and environmental control and monitoring.

The TK-80A is not only supported by our thorough documentation, it's backed by our 90-day warranty on the entire board and one year warranty on the components.

And the price is only $299.

At NEC Microcomputers, we've already built a reputation as one of the most reliable component suppliers in the industry. Now we're putting our reputation behind the first complete 8080A based computer on a board.

For more information on NEC's new college board, send in the coupon.



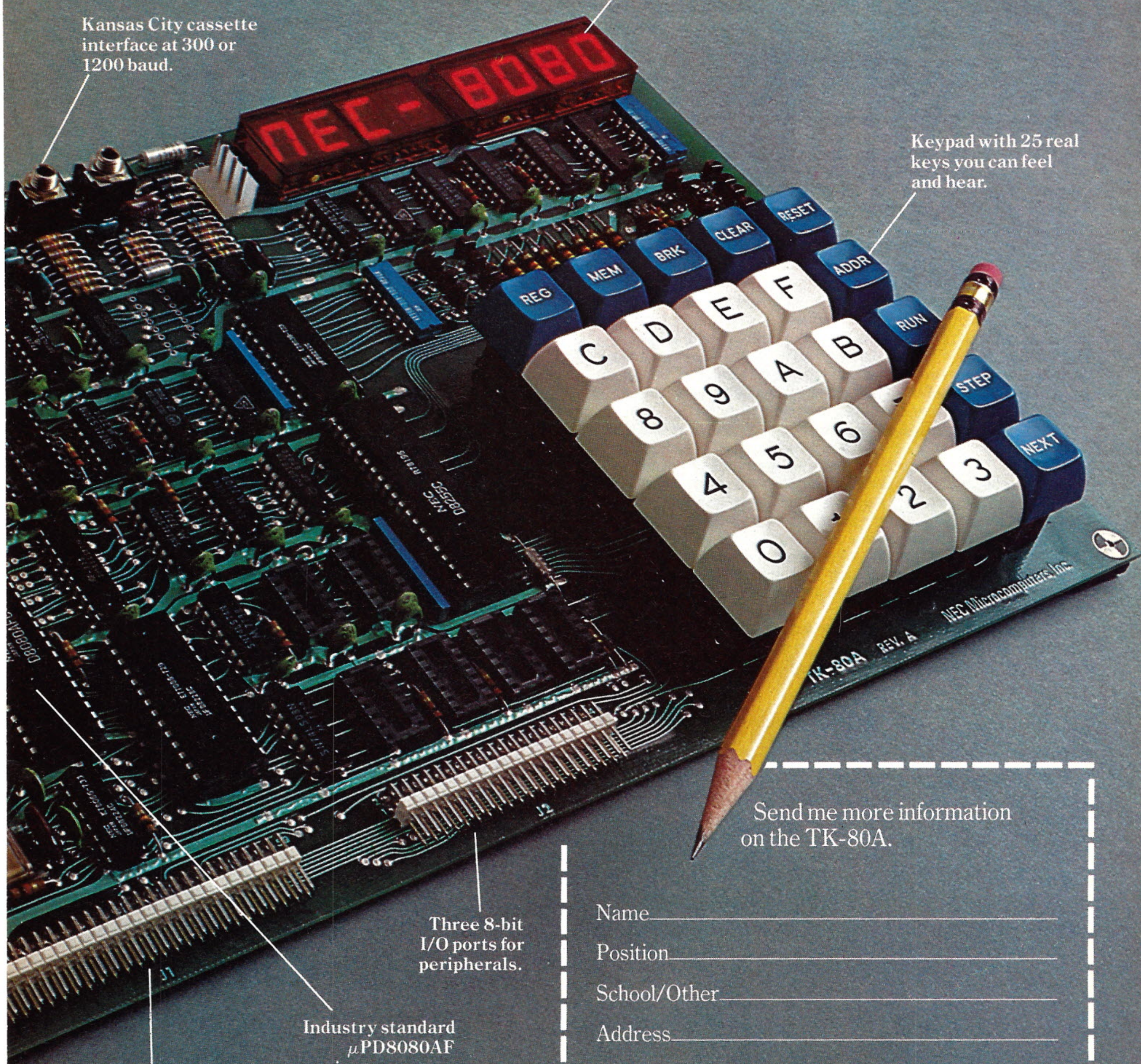
NEC

NEC Microcomputers, Inc.

WESTERN REGION OFFICE: Orange, CA at (714) 633-2980; EASTERN REGION OFFICE: Melville, NY at (516) 293-5660; MIDWESTERN REGION OFFICE: Des Plaines, IL at (312) 298-7081; SOUTHERN REGION OFFICE: Dallas, TX.

REPS: Action Unlimited, Arlington, TX, Spring, TX; Cerco, San Diego, CA; Contact Sales, Inc., Burlington, MA ; D/Z Associates, Inc., Denver, CO; Electronics Innovators, Inc., Minneapolis, MN; Eltron, Phoenix, AZ; Kaytronics Limited, Ville St. Pierre, QUEBEC, Concord, ONTARIO, Surrey, BRITISH COLUMBIA; Santana Sales, Costa Mesa, CA; Trident Associates, Inc., Sunnyvale, CA; Tri-Tronix, Albuquerque, NM; Tri-Tronix, NW., Mercer Island, WA.

DISTRIBUTORS: Almo Electronics Corp., Philadelphia, PA, Baltimore, MD; Bell Industries, Bellevue, WA; Century Electronics, Albuquerque, NM, Wheatridge, CO, Salt Lake City, UT; Diplomat/Westland, Inc., Sunnyvale, CA; Diplomat/Southland, Inc., Clearwater, FL; Diplomat/Lakeland, Inc., Elk Grove Village, IL; Diplomat/IPC of Mass., Chicopee Falls, MA; Diplomat, Holliston, MA; Diplomat/Northland, Inc., Farmington, MI; Diplomat/Electro-Com Corp., Minneapolis, MN; Diplomat/St. Louis, Inc., St. Louis.



Send me more information on the TK-80A.

Name____________________

Position____________________

School/Other____________________

Address____________________

City__________ State_____ Zip_____

Send to: NEC Microcomputers, Inc.
173 Worcester Street, Wellesley,
MA 02181 Tel. (617) 237-1910

CC-1

MO; Diplomat/IPC Corp., Totowa, NJ, Mt. Laurel, NJ; Diplomat Electronics Corp., Woodbury, NY; Diplomat/Alta-Land, Inc., Salt Lake City, UT; Future Electronics Corp., Montreal, QUEBEC, Downsview, ONTARIO, Ottawa, ONTARIO, Natick, MA; Hughes-Peters, Inc., Cincinnati, OH, Columbus, OH; Intermark Electronics, Sunnyvale, CA, Santa Ana, CA, San Diego, CA; KA Electronics, Dallas, TX; Kent Electronics, Houston, TX; G.S. Marshall, Sunnyvale, CA, Canoga Park, CA, El Monte, CA, San Diego, CA, Phoenix, AZ; Marshall Industries, Farmingdale, NY, Wallingford, CT; Milgray Electronics, Inc., Freeport, NY, Orange, CT; Quality Components, Austin, TX, Houston, TX, Dallas, TX; Reptron Electronics, Inc., Livonia, MI, Columbus, OH; Resco/Raleigh, Raleigh, NC; Semiconductor Specialists, Inc., Chicago, IL, Burlington, MA, Farmington, MI, Minneapolis, MN, Hazelwood, MO, Pittsburgh, PA, Dallas, TX, Milwaukee, WI; Sterling Electronics, Phoenix, AZ, Santa Clara, CA, San Diego, CA, N. Hollywood, CA, Baton Rouge, LA, Waltham, MA, Albuquerque, NM, Dallas, TX, Tukwila, WA; Summit Distributors, Inc., Buffalo, NY; Summit Elec. of Roch., Inc., Rochester, NY; Western Microtechnology, Cupertino, CA.

I/O, con't...

Dear Editor:

I feel compelled to protest both the blatant inaccuracies and the distortions concerning the Computer Mart of New Jersey as portrayed in your article entitled 'Mystery Shopper' in the June, 1979 issue. You have rendered a great disservice to both your readers and the industry by publishing these innuendoes and half-truths.

I do not wish to defend the store as I do not believe they are in need of my or any defense. They are, quite simply, THE BEST in their field. I have traveled extensively in the course of my business and visited several stores across the nation. I have found none as well-managed, helpful, friendly, intelligent, dynamic nor as honest. They are, at a minimum, one to two years ahead of their competition in anticipating the needs of their selling market and adapting to the changes in this market to continue meeting these needs. While others talk, they perform.

Perhaps the most important asset of any store to the more than casual buyer (and one totally overlooked in your article) is the quality of their service department. Our Company has never missed a single commitment to our clients as a result of computer malfunction. Our computer was repaired within one hour on a Saturday, traditionally their busiest sales day. While this may be the exception rather than the rule, I was assured that customers using their systems for business applications do receive some priority. I was also promised the use of their equipment should an emergency arise.

Finally, I wish to inform you of the multitude of man hours invested by Mr. Larry Stein, the President, in the design of my system. There were many evenings in which we remained past the closing hour to match their hardware with my company's requirements. The system was tailored to my needs after careful review of our methods of operation and was, happily, less money than we had originally budgeted. Mr. Stein felt that while more expensive systems would also perform as well, the extra cost was unwarranted. In some instances he felt the more expensive equipment was inferior to our selection.

If even one potential customer was dissuaded from entering the store on the basis of your article, they will have lost the opportunity to be served by the finest your industry has to offer.

Sanford E. Haber
Automated Accounting Associates
150 West End Avenue
Brooklyn, New York 11235

My goodness, Sanford, why are you over-reacting like this? The article, and the comments made by Patty Rust, were far from being as derogatory as your reaction indicates. Perhaps you should go back and read it again. However, as far as the Computer Mart of New Jersey (and particularly Larry Stein) are concerned, I'd have to agree with you...they're both top-notch. — *JTC*

Cheap Change

Dear Editor:

An At to Victor K. Heyman for his article on the IDS 225 Line Printer. I've had mine connected to my TRS-80 Level II since last March. I find it to be a super printer and I recommend it. I use it to prepare lesson plans for students and it is not uncommon for me to run off 40-50 copies of things I'm doing.

I do suggest that you can get around the expensive replacement of ribbons, however. I've found that I can remove and re-ink the inking rollers with ease (although it does tend to be a messy operation unless you're careful!). Remove the rollers and then drip inkpad ink on to each one until it is saturated, replace them, and you are in business. Rewinding the ribbon spools with any standard ½ inch ribbon is fairly easy. Total cost is somewhere around a $1.00 a change which is a far cry from the $12.00 for a factory change.

Bethany Prendergast
10129 Leisure Lane North
Deerwood
Jacksonville, Florida 32216

Apple Plugs

Dear Editor:

The comment by Steve North in his review of the ALF Music Synthesizer (June '79 issue) to the effect that that board is the only significant plug-in option for an Apple, is a little out of line. I'm certain that Mountain Hardware (Keyboard Filter, Super Talker, Introl), Heuristics (Speechlab), Talos (Digi-Kit-Tizer), D.C. Hayes (Micro Modem) and many others would take issue with it. At last count, our files listed more than 160 such companies and that number grows weekly.

I have always enjoyed your magazine because I find that it's light and informative style appeals to people who are not technical experts. But it is doubly important that your authors know what they are talking about.

Phil Roybal
Apple Computer, Inc.
10260 Bandley Drive
Cupertino, CA 95014

Believe it or not, the Apple is one of my favorite computers. All the boards you mentioned are noteworthy. **In Comparison with other personal computers,** *Apple ranks highly in terms of software and even hardware expansion. Perhaps I should have directed my comments at the entire personal computer industry.* — *SN*

Up With Apple!

Dear Editor:

I would like to clarify some comments of mine which were quoted by Steve North in his article on our Apple Music Synthesizer which appeared in the June, 1979 issue of **Creative Computing.** In particular, when I stated that it is difficult to design peripherals for the Apple, I meant that the Apple is not designed with ease of peripheral interfacing as a primary factor. A brief example is in order. The Hewlett-Packard 2100 series minicomputer is designed with peripheral interfacing as a **primary** factor. It will accept circuit cards larger than the Apple, it has more power available and its mighty cooling fans allow a great deal of heat dissipation. This results in two interesting side effects. It is the size and weight of a small refrigerator (especially the 2116) and is about as noisy. The second effect is at least one more digit in the price tag than Apple's. These are not acceptable in a personal computer. So Apple is designed with cost, size, weight, noise and appearance in mind. Note that expansion was not far down on the list. Steve's quote makes it sound as if I am picking on the Apple as being difficult to expand, perhaps even in contrast to other similar systems. In fact, the reverse is true. We picked the Apple to make a synthesizer for because it was (and still is) the only mass-market personal computer which would accept a plug-in circuit card. So who cares if the card isn't as big as a minicomputers card? (If you're thinking about S-100 systems being able to accept cards and maybe qualifying as "mass-market," we used to make S-100 synthesizers but it was too difficult to design software for all of the various systems, so we quit.) In all fairness to Apple, they must be complimented on their foresight in including eight expansion sockets and lots of empty cabinet space.

Philip Tubb
ALF Products
128 S. Taft
Denver, CO 80228

I'm sorry for any misunderstanding of your comments. Of all the personal computers, Apple is one of the best for expansion. Contrast all the Kluge boxes and cables needed to add a disk to a TRS-80, or the way OSI seems to discourage second-sourcing of hardware add-ons. But my own impression is that almost all personal computers are unnecessarily difficult to expand for one reason or another. — *SN*

$100 to $139 Off the TRS-80™ Computers!

You can believe a product is tops, when more than 100,000 owners have chosen it over its competition! That overwhelming acceptance has dropped production costs, and allowed us to make TRS-80 an even better value.

$100 Off! 4K Level I

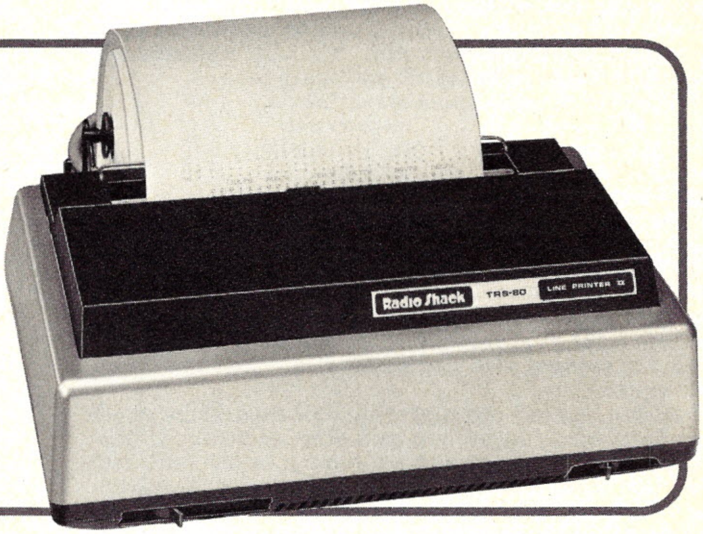
This is the ideal beginner's computer system. Our manual makes learning programming in Level I BASIC a snap. Easily upgrades to Level II or more memory at any time. **26-1051**
Was $599 in 1979 Catalog **$499***

$139 Off! 16K Level II

Level II BASIC is one of the most powerful microcomputer programming languages. Level II TRS-80 systems can be expanded to include printers, disk storage and much more. **26-1056**
Was $988 in 1979 Catalog **$849***

TRS-80 Line Printer II

Now—for hundreds of dollars less than you'd expect to pay—you can add line printer utility and convenience to your Level II TRS-80 system. This top quality impact-type printer doesn't require special paper—you can use inexpensive rolls (available at Radio Shack), continuous forms (original and up to two carbons) or single sheets. The 7x7 dot matrix head prints 50 characters per second; 80 characters on 8 inch lines. Also prints expanded (wider) characters that are ideal for headings under software control. Standard 9½ inch wide continuous forms are kept in perfect alignment by non-adjustable pins on platen. Includes Expansion Interface cable. **26-1154** **$999***



Radio Shack®
The biggest name in little computers™

A DIVISION OF TANDY CORPORATION • FORT WORTH, TEXAS 76102
OVER 7000 LOCATIONS IN NINE COUNTRIES

TRS-80 Line Printer II Interface Cable

Now you can attach the Line Printer II directly to your TRS-80 Level II 16K system. No Expansion Interface required unless you plan to add mini disks or more memory. **26-1416** **$59***

*Retail prices may vary at individual stores and dealers

Copyright Confusion

Dear Editor:

Since our agency recently purchased eleven microcomputers, we've been concerned and confused about reproduction of copyrighted software. Rather than guess or waiting for a confrontation, we're sending a letter to vendors asking about their policies. We ask four questions:

1. May we make copies to replace copies that become unusable?
2. Do you offer quantity discounts?
3. Under what circumstances, if any, may we make copies for use on our own agency's microcomputers. (Perhaps the vendor will give permission for a fee).
4. If we enhance one of your programs, under what circumstances may we make or allow copies?

We hope knowing answers to these questions will help us avoid/prevent some buyer-seller conflicts and help us decide what programs to buy.

Dick Ricketts
Multnomah County
Education Service District
Portland, OR 97216

Dear Mr. Ricketts:

Thank you for your recent letter. Our position is as follows:

Reproduction of copyrighted software is exactly the same as reproduction of a copyrighted book, i.e., it is not permitted without express written permission from the copyright holder.

In response to your specific questions:

1. We suggest that purchasers of Creative Computing software make one copy for daily use and save our original tape for backup. Our tapes have two copies of each program; one on each side for redundancy. We guarantee that they will load and run correctly if used in accordance with the written directions provided. We will replace tape cassettes or floppy disks that do not load or run.

2. We offer quantity discounts as follows:

1- 9 tapes	*no discount*
10-19 tapes (same title)	*20% discount*
20 + tapes (same title)	*40% discount*

3. We will license you to copy tapes (or disks) for your own use, but not for resale, for 20% of the list price of the tape for each unit copied. (If you make 20 copies of one tape or one copy of 20 tapes the license fee is identical, i.e., list price x 20% x 20 units).

4. Enhanced versions of the software are to be treated the same as the original. In other words, the copyright covers the original idea and concept, not just the exact words or execution.

I'm sure that you can appreciate that we have invested many hundreds of hours into each tape we produce. This involves time for programming, testing, evaluation and production. We are deeply committed to serving the educational market as well as the personal and small business markets. If school districts somehow figure that they are exempt from legal and ethical restrictions against copying software, we, as well as other software producers who are dealing with this market, will have no choice but to raise prices to compensate for low unit sales. We view this as undesirable. We would rather keep our prices low and attractive and put our software within reach of as many people as possible.

Very few software vendors are focusing on the education market today and two who were in it have recently withdrawn. The reason is low unit sales caused, partially, by the fact that school districts, under the plea of poverty, feel it is okay to copy software. This is a highly undesirable situation. We would like to see more competition in this field because it will broaden the entire field, and justify to more schools the purchase of microcomputer systems which will expand the market for everyone.

As a rule of thumb, we feel that a school contemplating the purchase of a computer should allocate a minimum of 20% of the total system price for applications software. Just as a microscope without a textbook, slides and ancillary equipment is useless, so is a computer. We want schools to get the most out of their computers — for this to happen the software developers will have to justify their investments and be convinced that this is a viable market. For example, we invested over $10,000 in the development of our series of Ecology and Economic Simulations. We feel that this is worthwhile but we have to sell over 1000 of each of these three tapes just to break even.

— David H. Ahl
Publisher

PTC Owners Don't Despair

Dear Editor:

By now you have probably learned all about the demise of Processor Technology. The users' group is still going strong, and indeed there is more of a need for the group than ever before. I plan to continue serving the needs of owners of Processor Technology Corporation products. We are lining up service centers throughout the US and in other countries, where former PTC dealers can continue to maintain the equipment with technicians trained at the PTC factory. Our software library continues to accumulate software on PTC media. We have accumulated an extensive archive of PTC documents, including software manuals, service manuals, etc. We plan to publish information on hardware and software updates that were obtained from PTC as they cleaned out their building.

There are many owners of PTC equipment who don't know of our existence due to the poor customer relations contacts of PTC. Any help you can give to spread the word would be beneficial to both us and them.

Stanley M. Sokolow,
Proteus
Executive Director
1690 Woodside Road, Suite 219
Redwood City, California 94061

In Search of Computer Portraits

Dear Editor:

I am interested in obtaining names, addresses and telephone numbers, if possible, of individuals and companies, who manufacture or design computer portrait systems (computer photography or image processing) for hobby and commercial use.

My interest includes companies offering turnkey systems, individual boards or subassemblies.

I would appreciate any help you can give me.

Hassan A. Mohamed
5701-8 Cedars East Ct.
Charlotte, NC 28212

Here are a few computer portrait companies for you to check into:

Computer Portraits, Inc.
2129 Hacienda Way, Suite J
Sacramento, CA 95825

Foto Fun, Inc.
189 Dean Street
Raynham, MA 02767

Computer Ventures, Inc.
P.O. Box 984
Acton, MA 01720

COMPIC Corporation
113 North Nell Street
Champaign, IL 61820

Computer Amusement Systems, Inc.
11 West 20th Street
New York, NY 10011

The Micro Works
P.O. Box 1110
Del Mar, CA 92014

Vector Graphics, Inc.
31364 Via Colinas
Westlake Village, CA 91361

DH Financial Associates
3906 Theota Ave.
Cleveland, OH 44134

Hope you find those useful. *—JTC*

Problems With "Sorting Simplified"

Dear Editor:

I'm embarrased to have received such wide circulation on a routine with a bug in it. However, my sort routine which appeared on page 78 of the July issue has a problem in that its sorting capabilities are sometimes affected by the type of data being sorted. (This problem was pointed out to me by Bob Doolittle in Pacific Palisades, California). Following is a correct listing of the program. My apologies.

Bill Roch
24000 Bessemer St.
Woodland Hills, CA 91367

```
START    REM    **************************
         REM    **                      **
         REM    **  SORT ROUTINE from   **
         REM    **                      **
         REM    **  BOB DOLITTLE        **
         REM    **  PACIFIC PALISADES   **
         REM    **  CALIF.              **
         REM    **                      **
         REM    **************************
         REM
         DIM W(50),U$(50),W$(50)
START1   INPUT "ENTER ID START POSN & ID LENGTH: ";S1,L1
         INPUT "FILE NAME: ";F$
         OPEN 2,F$
         FOR I=1 TO 50
         GET U$(I):     W$(I)=MID$(U$(I),S1,L1):        W(I)=I
         IF EOF(0)=-1 THEN GOTO ENDRD
         NEXT
ENDRD    G=I-1
         CLOSE 2
         GOSUB SORT
         FOR I=1 TO G:   PRINT U$(W(I)): NEXT :  STOP
         REM ******************
         REM *  SORT ROUTINE  *
         REM ******************
SORT     C=G
560      PRINT :PRINT "*** NOW SORTING ***":PRINT
570      C=INT(C/2)
580      IF C=0 THEN RETURN
590      J=1:    D=G-C
600      F=J
605      E=F+C
610      IF W$(F)<=W$(E) THEN GOTO 660
620      W$=W$(F):        W$(F)=W$(E):     W$(E)=W$
625      W=W(F):          W(F)=W(E):       W(E)=W
630      F=F-C
640      IF F<1 THEN GOTO 660
650      GOTO 605
660      J=J+1
670      IF J>D THEN GOTO 570
680      GOTO 600
         REM
         REM WRITTEN IN TARBELL BASIC
```

Help For The Deaf

Dear Editor:

I have a problem which I hope you would be able to assist me with. I plan on purchasing a microcomputer within the near future. I also plan on purchasing a phone communications system for my parents, who are deaf.

Thus, I wonder if I can program/hardware my computer in any manner so that I could talk to my parents over the phone. The main problem is that the deaf communication system has been standardized, using a now obsolete code, the Baudot, or level five. Of course, most modern equipment that I could purchase would not use this code.

I am therefore wondering if I can program my computer to simulate the Baudot code via software, or what hardware would be required to put both systems on the same plane. If you or your readers could offer any insight into this problem, I would be grateful and would endeavor to make the solution available to the deaf.

Alvin P. Dziurzynski, C.P.A.
26 South Morrison, Apt. #3
Memphis, TN 38104

Many amateur radio enthusiasts have developed programs for ASCII-to-Baudot code conversions [and vice-versa] because of the many Baudot machines being used by hams. There have been several articles on that topic published in past issues of **73 Magazine** *and/or* **QST Magazine** *(which you can find in a local ham radio shop). As a matter of fact, it seems that I've heard of a ham radio organization which has developed a communication system for the deaf. Perhaps a letter to the editors of either of those magazines would be helpful.* *—JTC*

Eliminate Fiction and Foolishness?

Dear Editor:

I'm 11 years old and think your magazine is the best. Before I started getting your issues I was out of touch with the world. Who believes an eleven year old won't break the computer system on display?

I'd like to comment on a letter written by Russell Sheley. He wrote that "Elimination of Fiction and Foolishness would be no loss." That's only **his** opinion! I happen to like that section. He doesn't have to read it.

Another thing, along with all the other things wrong with the Jan. '79 issue in the microchess review the author talked about pressing the return key. That's fine - only the TRS-80 doesn't have a return key.

If you plan to reprint the first computer games book may I suggest slashing all the zeros. I can't use some programs (like golf) because how am I supposed to know whether L(0) means L([zero]) or L(the letter 0)?

Bruce Klein

Academic User Groups?

Dear Editor:

Do you know of any user groups especially devoted to educational software in the fields of science, engineering and math? When I was in college the great love of my life was science but, alas, the mortal demands have gently eased me into the business world.

Now a great dream has been goading me ever since I heard of microcomputers two years ago. Can the gods have granted us knowledge-starved post-formal-educated adults private mini-universities for our own homes?

My dream has not been realized through any advertised software so I am appealing to you, the editors and readers of Creative Computing for information.

My computer sits idle while I wallow in books. It is waiting for the grad students and professors of the world to put their knowledge into the computer-age mode.

Think of all the hundreds of thousands of students who would benefit from this kind of software. And us, the work-a-day ex-scientists whose dreams of Schroedinger's equations and Fourier analyses and Laplace transforms ended because our four years were up and it was time to go out and face mere dreary dollars.

Gerald Waxler
Box 101
Spinnerstown, PA 18968

I haven't heard of such a user's group...but perhaps a reader has. I'd like very much to put together a list of all of the user groups in the personal computing field and publish it in the near future. *JTC*

Apple 3-D Graphics

Dear Editor:

Rich Milewski just scratched the surface with his great article on true 3D graphics. I wish my IMSAI and TRS-80 had color so I could join in the fun of blasting my CRT to pieces with phaser fire!

Deep Vision Corporation, 6110 Santa Monica Bl, Hollywood 90038 has real movie-type 3D glasses at $1.00 each.

Stephen Gibson
547 N. Beachwood Dr.
Los Angeles, CA 90004

MOVING DATA AT A SNAIL'S PACE BECAUSE YOU'RE FLOPPY BOUND?

Let Corvus Systems put you back in the race!

- **For TRS-80†, Apple‡ (including Apple Pascal), S-100 Bus—and now LSI-11.**
- **Fully compatible hardware/software.**
- **10-million byte disk: IMI-7710.**
- **Proven Winchester technology.**
- **Z-80 based Corvus disk controller.**
- **Comprehensive disk diagnostics.**
- **Up to 4 disks per system.**
- **System $5350, add-on disk $2990.**

Corvus offers a complete systems solution to the mass storage problem of micro computers. In a package smaller than a briefcase, we provide an intelligent controller, disk, and personality module. Call or write today for additional information. Get up to speed with Corvus.

NOW! CORVUS SPEAKS APPLE™ PASCAL™!

†TRS-80 is a registered trademark of Radio Shack, a Tandy Co.
‡Apple is a registered trademark of APPLE Computers, Inc.

CORVUS SYSTEMS, Inc.
900 S. Winchester Boulevard
San Jose, California 95128
408/246-0461

CIRCLE 226 ON READER SERVICE CARD

Notices

4th Computer Swap Meet

It's time for good times and good deals at the largest swap meet for computer enthusiasts on the West Coast. Whether you're shopping for an entire system or an integrated circuit, a disk drive or diskette, an applications program or game, a book or magazine, or whatever..,you'll find it at the 4th Annual California Computer Swap meet. However, before you come out as a buyer, perhaps you should check over that garage or back room and decide if it's time to unload some of that "junk" you've been accumulating...and get it into some other guy's garage! (That last applies to computer stores and large companies as well as individuals.)

The 4th Annual California Computer Swap Meet will be held on September 15th, from 9 to 5 p.m., at the San Mateo County Fairgrounds...just a short drive south of downtown San Francisco on Hwy 101. If you're going to be traveling to the Bay Area be sure to plan on making a weekend of it...there's a lot to see and do.

This event has become bigger and better each year with a wide variety of items being offered for the popular personal systems...both for the hardware and software enthusiasts and end users. With this year's swap meet being held in the center of the Bay Area's computer/ electronics industry it should be a real blockbuster.

Buyers and sellers will be coming from the entire West Coast. Admission is free to buyers (with the exception of a $1 Fairgrounds parking fee). Sellers, both individuals and companies, are invited to call John Craig, Editor of Creative Computing Magazine, at 805-735-1023 for booth prices and availability (or write to: RFD Box 100 D, Lompoc, CA 93436). It's going to be a BIG event. There will be notices sent out to all computer stores, clubs and manufacturers in the Western U.S. and press releases should appear in practically every personal computing magazine and newsletter as well as several professional publications.

Having a good time at this event is not optional...it's mandatory!

World Power Systems Update

Norman Henry Hunt, alias Jim Anderson and Col. David Winthrop, was captured on May 26 in Honolulu. He is scheduled for trial in federal court in Los Angeles on July 17 on 14 counts of mail fraud stemming from the Data Sync caper in 1977. Following that, he is scheduled to be tried on an unknown number of mail fraud counts in Tucson for the World Power Systems SCAM. He will then return to prison to finish serving the sentence he was serving when he escaped, and the state will prosecute him for that escape.

He is also facing charges in Texas for allegedly swindling a federal judge. If the microcomputer industry is lucky, he should be out of commission for about 10 years.

More March Mix-ups

There were a few typographical errors in the PET column in our March 1979 issue. Here are the corrections.

In the Scott Joplin Music Player program, the line number following 110 should be 120, not 200. There should be a comma in line 260 so that it reads: 260 IF MID$("AAB CCDDEFFGG", J, 1).

In the Transition Music Player, make these corrections:

```
60 IF N<0 THEN 100
140 L = LEN(C$(N))
```

Apple Education Foundation

Recognizing the need for expanded educational opportunities, Apple Computer, Inc. today announced the formation of the Apple Education Foundation. Initially funded by Apple Computer, the nonprofit foundation will offer support and resources to organizations and individuals who are pioneering learning methods through the use of microcomputers.

The foundation will distribute hardware equipment for both developmental and demonstration projects involved in producing instructional computing materials. In addition, a few funding grants will be available for educational enrichment projects.

Final grant proposals and authorizations for funding disbursements will be reviewed by a board of directors, backed by an advisory council composed of leaders in the field of computer-based education. The advisors will provide guidance, and will review grant applications and submit them for final approval by the board of directors.

The foundation's primary goal is to place hardware into the hands of people who will further those educational methods which take best advantage of the personal microcomputer's capabilities.

The foundation will also sponsor the Education Program Information Center (EPIC). EPIC will support microcomputer users in developing new instructional programs and in obtaining available information on educational materials. The center will publish information packages containing design and development guides, editorial and marketing guidelines, software techniques and authorizing tools. Authors are encouraged to submit their work to the center for review and feedback on the most effective uses and placements of their materials.

Further assisting microcomputer users, EPIC's Users Guide will give overviews of state-of-the-art computing, plus critical reviews of educational programs available for popular small computers.

Both the Apple Education Foundation and EPIC may be contacted at:

Apple Education Foundation
20605 Lazaneo Drive
Cupertino, CA 95014

To All Programmers

Before buying a single additional tape cassette, magazine, or programming text, go out and buy a copy of Webster's New Collegiate Dictionary (or any other dictionary for that matter). In it, you will learn (hopefully):

1. Congratulations is not spelled congradulations.
2. Your does not mean the same as you're which is a contraction for you are.
3. Lasers are a device for annihilating people who spell the word "lazer."
4. Numbers over 10,000 use commas to group digits into units of three, but numbers under 9999 are generally not divided.
5. Exclamation points are most often required when you can't find an appropriate word to express your thoughts. We don't like them.
6. A double hyphen (or dash) is used for entirely different purposes than a hyphen. Use them correctly and then you can take pride in your work. (Notice too, that "than" is not equal to "then." And that "too" is to be differentiated from "to" and "two." Also that sentences may occasionally begin with a conjunction.)
7. "Can" means am (is, are) able. Not to be used as a substitute for "may."
8. "Data" are many facts or numbers. The word is a plural, like "phenomena" and "strata." One piece of data is a datum.
9. "Effect" is a noun meaning "result;" as a verb it means "to bring about" or "accomplish." It is not to be confused with "affect" which means "to influence" and is always a verb.
10. "A number of" is horrible. A much better choice would be "four," "30,000," "several," "scores," or "a few."

Although I hail from Cornell and thus feel that William Strunk and E.B. White are folk heroes, there's something for everyone in "The Elements of Style." Or look up some of the writing of Edwin Newman or William Safire. And please stop filling our mailbox with ungrammatical, multilations of the English language. As Winston Churchill said, "this is something up with which I shall not put."

— DHA

Our Face Is Red

Contrary to the subheadline, "Brain Teaser" in the July issue, page 104, was not "Another new game from Creative Computing." We find it was previously published by Willard Nico in the May 1976 issue of **Byte** under the name "Shooting Stars," and also in People's Computer Company's book **What to do After You Hit Return**. We apologize to Mr. Nico, PCC, and our readers.

Intel University Program

The fast pace of electronics makes it important for new engineering graduates to be conversant with microcomputers, memories, and programming. In addition, experienced engineers may require refresher courses on the latest technologies. Intel supports colleges and universities with a variety of special programs and discounts.

Three sets of microcomputer components are available for laboratory or research use. They are made up from visually imperfect but functional IC's.

One set is based around the 8086 16-bit MPU, one around the 8080A and a third set is based on the 8035. These sets are available at prices between $20 and $90. **Colleges and universities only** may write for further information to Rob Walker, Manager, Marketing Communications, Intel Corp., 3065 Bowers Ave., Santa Clara, CA 95051. (Mention Creative Computing).

Conference

On October 21-23, 1979, the NY State Association for Educational Data Systems Annual Conference will be held at the Granit Hotel, Kerhonksen, NY. The theme is Instructional Computing-Hardware/Software/Courseware. For further information contact Ms. Mary E. Heagney, 9201 Shore Road, Brooklyn, NY 11209, (212) 596-5850.

Apple Cart 3-D Graphics

Thanks to Al Booth of San Jose, CA for bringing July Apple-Cart's missing program to our attention. And here it is:

```
10  REM  HIRES GRAPHICS IN 3D
20  REM
100  HGR : POKE  - 16302,0
500  HCOLOR= 3
505  FOR X = 10 TO 270 STEP 50
510  HPLOT X,10 TO X,180
520  HPLOT X + 1,10 TO X + 1,180
530  HPLOT X - 1,10 TO X - 1,180
590  NEXT X
1000  FOR K = 1 TO 6
1010 XL = K * 20:YL = XL * .7
1020 IP = 6 - (XL / 30)
1025  IF K / 3 =  INT (K / 3) THEN
IP = 0
1030  GOSUB 8000
1040  NEXT K
1999  END
8000  REM  COLOR SEPARATIONS
8010  HCOLOR= 2
8020  IF IP = 0 THEN  HCOLOR= 3
8025 X = 140:Y = 96
8030  GOSUB 9000
8040 XL = XL + 1
8050  GOSUB 9000
8100  IF IP = 0 THEN  RETURN
8110  HCOLOR= 1
8120 X = 140 + IP + 2
8130  GOSUB 9000
8140 XL = XL - 1
8150  GOSUB 9000
8999  RETURN
9000  REM  PLOT THE DIAMONDS
9030  HPLOT X - XL,Y TO X,Y - YL
9040  HPLOT X + XL,Y TO X,Y + YL
9050  HPLOT X - XL,Y TO X,Y + YL
9060  HPLOT X + XL,Y TO X,Y - YL
9999  RETURN
```


The Greatest Shows On Earth

Betsy Staples

Consumer Electronics Show

"I guarantee that it's bigger than any show you've ever been to." That's how Publisher David Ahl summed up his description of the Consumer Electronics Show as we drove from Chicago's O'Hare airport to McCormick Place where the show was being held. He was right.

Although a relative novice in computerland, I had attended several personal computing shows, including the West Coast Computer Faire two weeks earlier and Computer Fair 3 at the University of Wisconsin—Parkside the day before. But, I was not prepared at CES for exhibitors' booths the size of houses, bevies of semi-clad women or a crowd of nearly 50,000 attendees on the first day.

Relaxing with some good music at the KLH hospitality suite.

Since Creative was not exhibiting at the show, we had time to inspect the three levels of exhibits, two of which were larger than any show I had attended before. Smokey lucite and chrome, large plants and thick carpeting were mainstays of the decorators, and most of the booths had nicely furnished private meeting areas where sales people could meet with prospective dealers.

Personal computing was obviously not the focus of this show, but even a newcomer to the industry could tell that Atari, Ohio Scientific, Cybervision, Interact, Exidy and others were making a place for themselves among

the manufacturers of sound systems and supplies, calculators, watches and pornographic video cassettes.

In fact, the main topic of conversation among those with whom I spoke was the new home computer introduced by Texas Instruments, the TI-99/4. Although the general consensus seemed to be that there was nothing particularly innovative about the machine, my first impression was that it was a computer even I could use without too much difficulty. The color monitor, nifty graphics and emphasis on educational applications were features that made it particularly appealing to me.

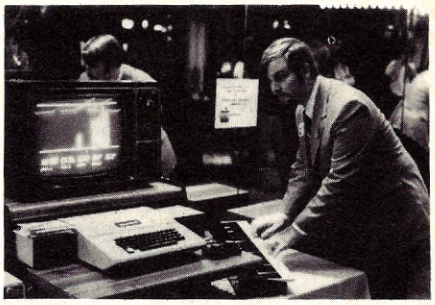
Mattel's Intellivision also appeared to have some fairly serious educational programs, but it was impossible to tell for sure, since the instructionless demonstrations left entirely too much to the imagination.

For those lucky enough to get close to Atari's demonstration games and computers, there were several new cartridges to test, at least one of which was also in need of some instructions. (If master game player and superhero fanatic David Ahl can't figure out how to play "Superman," there is little hope for the rest of us.)

Cybervision was displaying a very cute storybook type program. A voice on audio tape narrated "The Empress and the Nightingale" as illustrations appeared on the screen and occasionally requested input from the listener/viewer—the dawn of a new era in storytelling. "Mom, may Cybervision tell me just one more story, please!"

At the Interact booth I learned how long it was going to take me to pay off the second mortgage on my house, and later at their hospitality suite, I reviewed an imaginary stock portfolio in which I could change any of a number of variables and find out how my profits would be affected.

Conspicuously absent from the exhibit area was Apple. Although they had a suite in one of the nearby hotels, they had apparently decided to use most of their resources to impress a selected few dealers and members of the press at an elegant gathering in the Playboy Club. Guests, who were invited by a four-verse singing telegram, were treated to bowls of shrimp and crab claws, fresh fruit and cheese, delightful finger sandwiches, and assorted hot hors d'oeuvres while bunnies plied them with drinks from the bar.

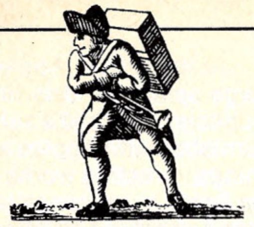


"The Entertainer" at Apple's posh Playboy Club reception was Charles Kellner.

Press kits, catalogs and other information were available in abundance, but the hit of the party was Charles Kellner with his musical Apple and its two ALF music synthesizer boards. The only thing that detracted from the elegance of the evening was a drawing conducted by the bunnies for a number of tacky Playboy prizes which seemed rather unprofessional and inappropriate.

The main emphasis of CES, however, was not computers, but hi-fi and stereo equipment, and the word that stands out in my mind when I think back on that part of it is "volume." As one who knows virtually nothing about sound equipment, two days at CES convinced me that the main criterion applied in deciding which equipment to purchase must be an assessment of its ability to produce more noise than others of its ilk.

From the seven-foot tall imitation bearskin covered speakers demonstrated by Fosgate to Advent's Sound Space Control, the objective of all the manufacturers seemed to be to produce as much noise as possible.



There was also distressingly little variety in the music chosen to demonstrate the noise-making capacity of the various systems. With few exceptions, contemporary, disco-type music blared incessantly. Only twice in two days did I hear classical selections that made me want to linger and listen. One of these was in KLH's hospitality suite in the McCormick Inn—a quiet, tasteful oasis where show goers could relax and enjoy refreshments and a variety of music from Brahms to Olivia Newton John emanating from amazing 11-inch cubes, the KLH Model 3 with computerized control.

Electronics is obviously still a male-dominated industry. There were very few women in evidence in other than decorative or receptionist capacities, and as a woman attending the show, I found myself being ignored by most of the exhibitors even though I had serious requests and questions. I usually had to walk straight up to someone and ask for help before my presence was even acknowledged.

This refusal to take women seriously was, of course, fostered by manufacturers' and show management's decision to use women's bodies as a means to attract attention. Playboy bunnies, Penthouse pets and other assorted females in tiny shorts or evening gowns left the distinct impression that women's most important role in the industry was as an attention-getting device.

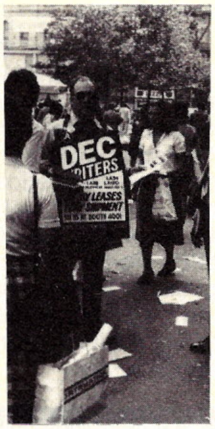
A refreshing variation in the attention-getting game was the show put on by the Kraco parrots who peddled a bicycle, dribbled a basketball and spoke on a Kraco CB radio. Between acts, their trainers gave away fluffy miniature parrots which clung as tenaciously to T-shirts as to suit jackets.

Parker Brothers, Coleco and Milton Bradley were demonstrating the same electronic games and toys they introduced at the Toy Fair in New York in February, with the significant addition of Milton Bradley's game cartridges for the TI-99/4.

Of course, there were hundreds of other exhibits of interesting and not-so-interesting, new and not-so-new products, but they have long since merged into a chrome and lucite, multi-dialed, LED flashing blur.

National Computer Conference

Back on the East Coast, Creative Computing was well represented at the National Computer Conference in New York City with two booths, one in the professional area and one in the personal computing section — an exhibit in the NCC "Hands-On" room, and a number of employees participating in the Conference.



Roller Skates were one means of transport in the Big Apple. Would you lease a DECwriter from this man?

Publisher David Ahl and Senior Programmer Ann Corrigan presented papers in a session on computer games and simulations; Editor John Craig participated in the judging of non-commercial demonstrations and chaired a session entitled "The Personal Computer in the Schools;" Managing Editor Burchenal Green served as vice chairwoman of the Steering Committee for the Personal Computing Festival and chaired a plenary session on The Coming Small Computer Revolution; and all employees were called upon to staff the two booths.

Like CES, NCC was quite different from most of the personal computing shows at which Creative exhibits. The professional computing people in their three-piece suits were a marked contrast to the T-shirted hobbyists we usually see, but clothes don't make the computer buff, and beneath their conventional exteriors, the people we met at NCC were just as enthusiastic about space games and Creative Computing as the hobbyists.

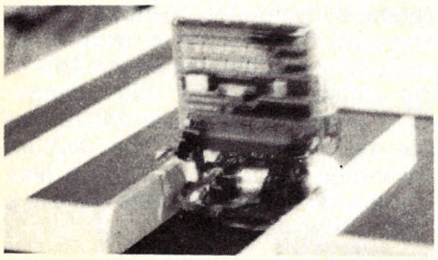
Souvlaki with onions or a dance.

The New York show, although not really larger than CES was more spread out with professional exhibits on four levels of the Coliseum, personal computing exhibits in the Sheraton Center, and a mixture of professional and personal exhibits in the Hilton. Although this arrangement made it a bit more difficult to see everything, it had the distinct advantage of providing a much less sterile atmosphere than one finds in a "convention center."

Sidewalk musicians give people a good sendoff on their journey from the Coliseum to the Sheraton.

For example, the matter of food. In Chicago there was an expensive and extremely limited selection: hot dogs or Italian sausage in McCormick Place or a roast beef sandwich in the McCormick Inn. Right outside the doors of the Coliseum, however, was a flock of vendors offering souvlaki, orange juice, pretzels and soda. And if street food didn't appeal to you, there were many other choices — from Burger King to some of Manhattan's finest restaurants — within easy walking distance.

Shows, con't...



An electronic mouse feels its way around the maze in the IEEE Spectrum/Computer competition.

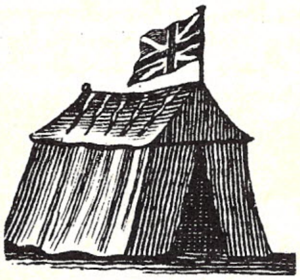
Speaking of Manhattan's finest restaurants, the NCC Committees sponsored receptions at two of them. On Wednesday evening, Personal Computing Festival Committee members mingled with luminaries from the personal computing community at the Stork Club. Waiters in tuxedos passed hot and cold hors d'oeurves to such well known guests as Creative Computing contributing editors Tom Dwyer, Margot Critchfield, Lee Felsenstein, Gregory Yob, Ted Nelson and Steve Gray.

Creative Computing had two booths at NCC, one in the professional area and one in the personal computing area.

The personal computing exhibit area was constantly busy.

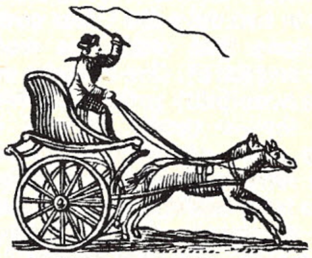
Windows on the World, on the 107th floor of the World Trade Center, was the site of a reception for Committee members and their guests on Thursday evening. Here the waiters didn't wear tuxedos, but the management was certainly fussy about the attire of the guests. Gloria Vanderbilt's signature notwithstanding, this guest was denied entry to the elevator until she exchanged her jeans for a friend's shawl (turned into a wraparound skirt).

The NCC "Computers in Home Education" session was packed to overflowing.



On the show floor, the personal computing area was filled with familiar faces from Exidy, Radio Shack, Jade, Ohio Scientific, Heath and other regular exhibitors. The professional area proved much less familiar. The monstrous exhibits of IBM, Univac, Honeywell, AT&T and other giants were crowded with people who had a much deeper understanding of and appreciation for large computers than I do.

At the Hilton, Apple again made a large splash — this time a bit more publicly. They had a suite, elaborately furnished with Apples all doing clever things: creating works of art, turning on lights in response to an oral command, teaching children, engaging in scientific research and—surprise!—making music. (Mr. Kellner must by now be able to play "Bugler's Holiday" and "Jesu, Joy of Man's Desiring" in his sleep—if, indeed, he ever gets any.)



Early personal computing proponent Ted Nelson and Abbie Gellis, President of the New York Area Computer Club, chat at The Stork Club during NCC.
Ted Nelson wasn't giving away any secrets about his latest projects — but watch these pages! Photo by Joe Kasser.

Following the show, a New York radio station reported that 79,000 computer enthusiasts from all over the world made this year's NCC the largest four-day festival to which New York City had ever been host. Next year it's back to the West Coast for NCC—we'll see you in Anaheim. ☐

Expand your TRS-80.
Save $100.

Meet the Vista V80 Mini Disk System. The perfect way to upgrade your TRS-80* system. Inexpensively. (Our $395.00 price is about $100.00 less than the manufacturer's equivalent.) Here's how it can help you.

23% more storage capacity. Useable storage is increased from 55,000 to 65,000 bytes on drive one.

8 times faster. While electronically equal to the TRS-80 Mini-Disk system, track-to-track access is 5ms versus 40ms for the TRS-80.

Better warranty. The V80 carries a 120 day warranty – longer than any comparable unit warranty available.

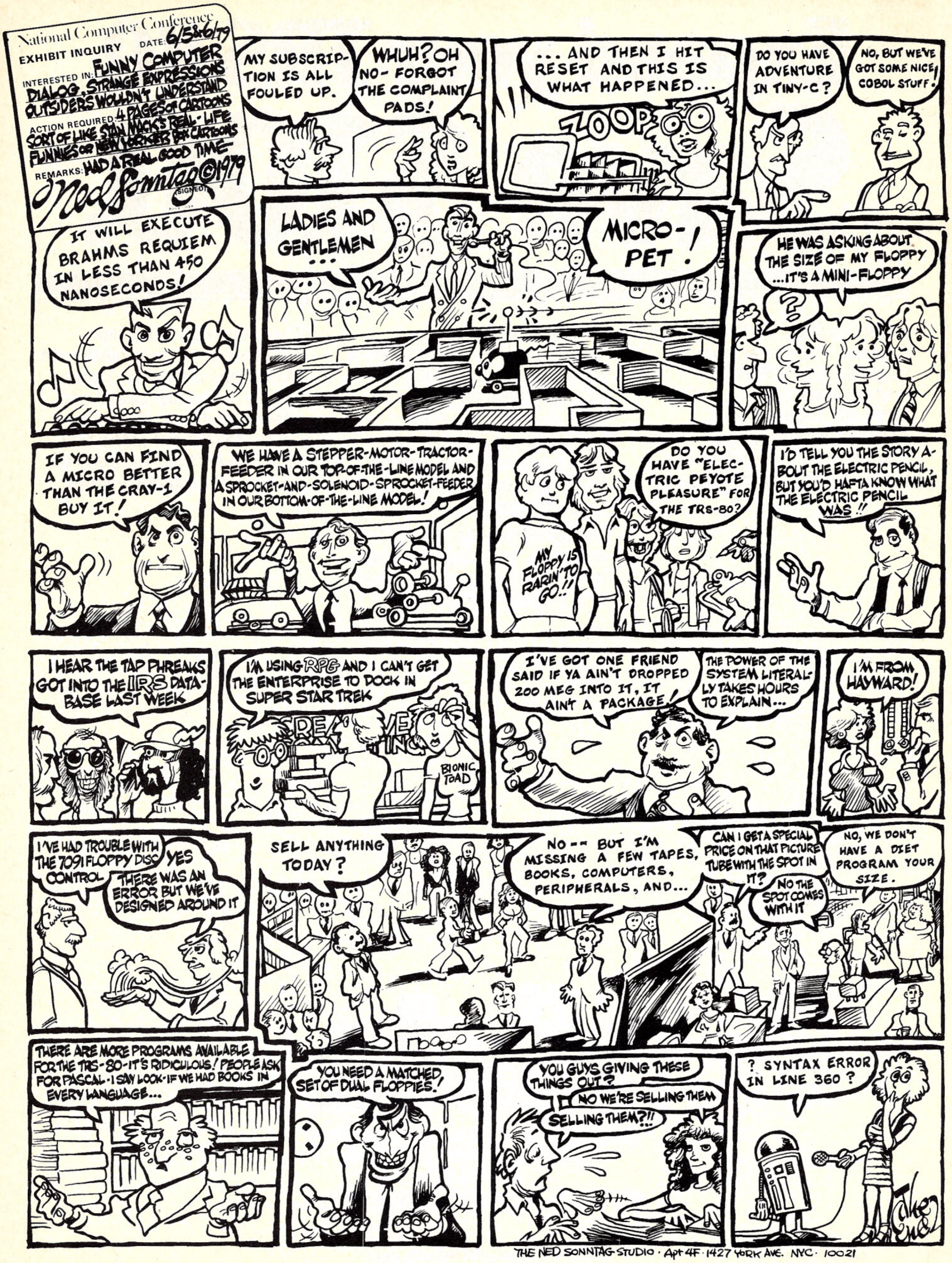
The Vista V80 Mini Disk System comes complete with Minifloppy disk drive, power supply, regulator board and case. And it's ready to run – simply take it out of the box, plug it in and you're ready to go. Dealer inquiries invited.

Vista

1320 East St. Andrews Place, Suite I
Santa Ana, California 92705
(714) 751-9201

At Vista, we mean business.

CIRCLE 211 ON READER SERVICE CARD



THE NED SONNTAG STUDIO · Apt 4F · 1427 YORK AVE. NYC · 10021

Get Ready to Make Your Own Magic

\nnouncement I. The first eight Personal ?rograms® from Aladdin Automation are vaiting for you now at your neighborhood :omputer retailer or direct from Aladdin.

\low you can get your full share of Aladdin nagic in every one of these Personal ?rograms®:

M**ath-Ter-Mind®** A delightful, educational learning experience for your pre-school child. Watch ne smile on your child's face as a correct nswer makes the mathematician smile on the creen before you. A nursery song also serves s a reward for learning elementary addition nd subtraction. With Aladdin's Math-Ter-Aind® your child's pathway to learning will be in-filled . . . for both of you. Math-Ter-Mind®. ne first release from the Aladdin Education® eries. (nursery song currently available only n Apple II® program)

L**unar Lander** In a controlled descent, you're just seconds away from your first landing on the cold, rbidding surface of the moon. As you avigate your delicate spacecraft downward to ie safety of Moonbase, you must be ever atchful of the dangers rising to meet you with ach passing moment: a fuel level fast pproaching zero; deadly meteor showers that ome from any direction, at any time; sheer-ced rock cliffs and rough terrain; choosing e correct landing pattern and rate of descent. addin's Lunar Lander. Your chance to reach ut and touch the stars . . . without leaving the ifety and comfort of your own chair. The first lease from the Aladdin Simulation® Series.

C**raps** All eyes in the casino are on you. The dice are in your hands. Lady Luck sits at your shoulder, whispering . . . "Just one more time. Try your luck just one more time." You throw . . . and watch the dice tumbling on the screen. With Aladdin's Craps you play against the computer, so it's awfully tough to win. But when you do, it's an experience you're likely never to forget. Craps. An exciting, heart-pounding Personal Program®. The first release from the Aladdin Las Vegas® Series.

M**astermind** A challenging game of intrigue, centuries old, that will give you full chance to test your powers of logic, deduction and reason. And test them you will, as you try and solve the computer's puzzle, using clues as they're provided one-by-one. You control the degree of difficulty in this classic Personal Program® that offers one simple, yet all-consuming challenge: beat the Mastermind in a direct, one-on-one battle of wits. Aladdin's Mastermind. The first release from the Aladdin Old Favorites® Series.

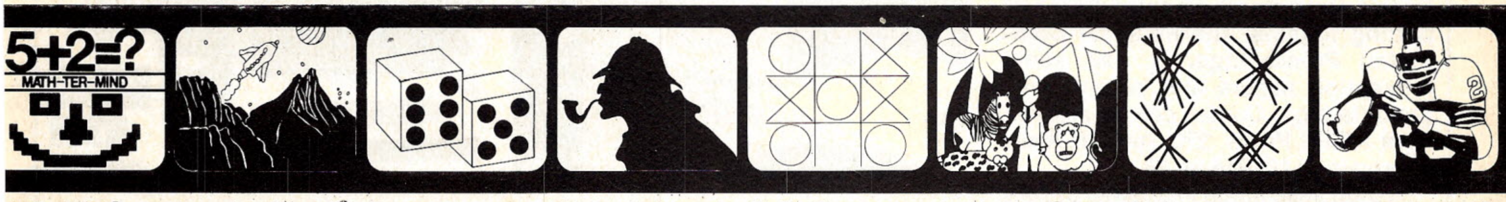
T**ic-Tac-Toe** Five different levels of difficulty allow a person of any age or skill to take part in this relaxing, enjoyable game that can act as a learning tool, as well. Level I, for example, is suitable for children and is excellent also for teaching simple mathematics. The computer plays just about perfectly at Level V. Just about, that is, so go ahead and take your best shot. See if you can beat the computer in this traditional favorite of young and old alike. Tic-Tac-Toe. Another first release from the Aladdin Old Favorites® Series.

J**ungle Island®** Shipwrecked in a raging storm at sea, miraculously you survive only to find yourself stranded on a seemingly deserted jungle island. Without food, water or supplies of any kind, you begin to try and find your way to safety. The computer will be your eyes and ears as you explore your jungle island and all the mysteries and dangers that lie in wait for you. Jungle Island®. A captivating first release from the Aladdin Adventure® Series.

S**tix®** Aladdin's Stix® can be played with 2 to 5 piles of sticks and between 1 and 19 sticks in each pile. The object: to be the one to pick up the last stick. Sounds simple? Yes, but you're playing against the computer. Take heart, though, because you can control the degree of difficulty in this update of the ancient game of Nim. Stix®. Another first release from the Aladdin Old Favorites® Series.

S**uper Pro Football®** Here's your chance to be more than just an armchair quarterback. With Aladdin's Super Pro Football® you can replay any Super Bowl game, from the first, between Green Bay and Oakland, to last year's classic victory by Pittsburgh over Dallas. For once you can turn back the clock and go for that one big play that made the difference between victory and defeat in pro football's biggest game of all. Super Pro Football®. The first exciting release from the Aladdin Super Pro® Series.

Visit your neighborhood computer retailer or contact Aladdin direct to get your full share of the magic in Announcement I, the first eight Personal Programs® from Aladdin Automation.



ath-Ter-Mind® Lunar Lander Craps Mastermind Tic-Tac-Toe Jungle Island® Stix® Super Pro Football®

'elcome to the All-New World of laddin. And Get Ready to ake Your Own Magic

ALADDIN AUTOMATION, INC.
A DIVISION OF ALADDIN COMPUTER CORP.
3420 Kenyon Street, Ste. 131, San Diego, CA 92110

Design and copy by Campbell Marsh Graphic Communications

CIRCLE 105 ON READER SERVICE CARD

NCC Personal Computing: Getting Bigger and Bigger

John Craig

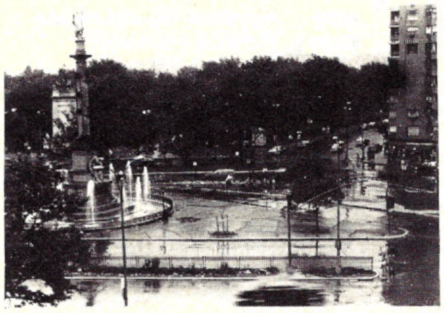
This year's National Computer Conference was held in the New York Coliseum, next to Central Park in Manhattan, from June 4th thru the 7th. The personal computing exhibits and seminars, in the Sheraton Centre Hotel, were bigger and better than ever. It's getting harder for the "big guys" to miss us at these shows...although sometimes I wonder if we really want them to notice too much. I'd prefer they continue doing what they do and not get any ideas about jumping in this market.

Perhaps I don't need to worry about the "big guys." The personal computing exhibitors weren't even mentioned in the Official Exhibit Guide...nor were any of the personal computing sessions mentioned in the seminar schedule (and the guide was 116 pages long). There were "editorials" in the front of the guide, written by three chairmen of the conference and the Executive Director of AFIPS, and only one of them made a single passing remark about the personal computing exhibits. Heck, let 'em keep their blinders on, right?

The Personal Computing Steering Committee is to be congratulated for the fine job they did this year. They were: Richard Kusmack, Chairman; Burchenal Green (of Creative Computing), Vice Chairwoman; Russel Adams and Jay Lucas, Program Directors; Edward Fox and Joe Kasser, Demonstrations, Charlie Floto, Communications; and Harriet Shair, Operations. Jay Lucas and Russ Adams also did a splendid job of putting together the Personal Computing Proceedings. It has some interesting papers and several program listings. Copies can be ordered from AFIPS Press, 210 Summit Ave., Montvale, NJ 07645 for $8.

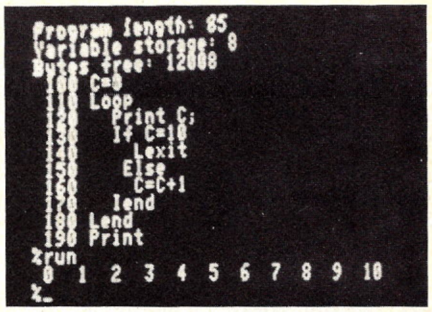
I didn't see much in the way of new hardware at NCC...it seems that most of the new systems were introduced at the 4th West Coast Computer Faire in May. There were a few new software and hardware packages from various companies but I felt the individual exhibits were some of the most interesting parts of the show (and I'll share most of them with you in the accompanying photos).

See you in Sunny Southern California next year!

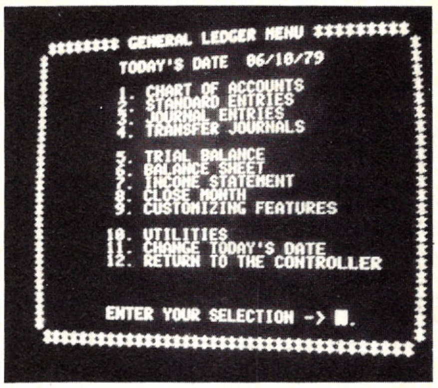


NCC '79: Overlooking Central Park and Columbus Circle from the New York Coliseum.

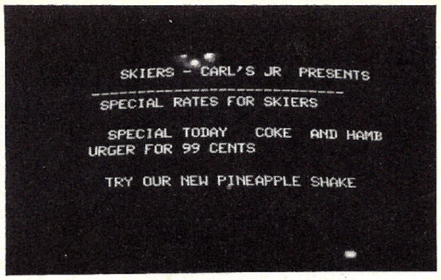
Apple is on the move! They introduced several new hardware and software packages at NCC. If I'm not mistaken, that's David Ahl enjoying the sights and especially sounds coming from their new music generation program (some really nice sounds there). However, the most significant happening was Apple's plunge into the business market. They introduced a General Business System, called The Controller, which consists of a General Ledger capable of handling up to 250 accounts, a 250-customer Accounts Receivable and a 100-customer Accounts Payable module. In addition, they were demonstrating an inventory control and cash register simulation system for retailers, called the Cashier, and a mailing list maintenance program called Apple Post. The main menu from The Controller is shown in the center photo with a typical hardware configuration to the right. For additional information send $2 for a copy of "Computers in Business" from Apple Computer Inc., 10260 Bandley Dr., Cupertino, CA 95014.



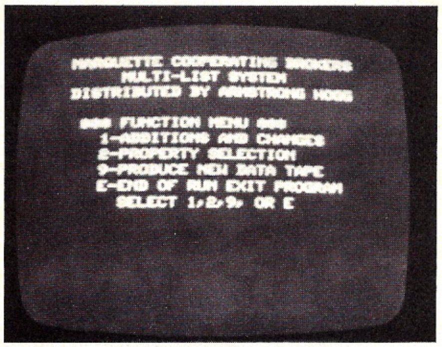
Glen Roberts and Steve Andre of Ann Harbor, Michigan 48104 (1509 Kearney and 1605 S. University, respectively) have developed a structured BASIC which has LOOP, WHILE and IF THEN ELSE control structures. It can perform some very fancy footwork (because of its many features) and anyone looking for an 8080/Z80 BASIC to market should give them a call.



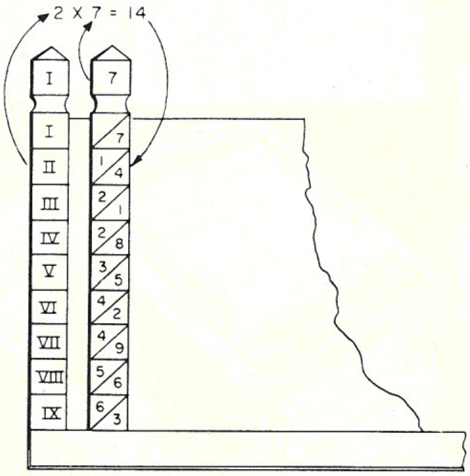
NCC, con't...



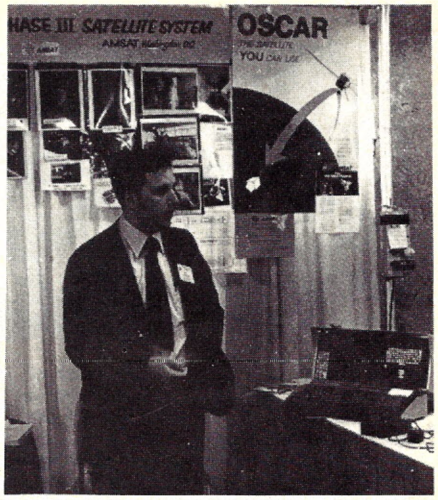
You ski enthusiasts, especially those in California, are going to love this one! Preston Janes, the gentleman in the left photo, has developed Something New Under The Sun," called Skicom. The system, which uses TRS-80s as remote terminals, provides up-to-the-minute ski reports from resorts in the Southern California area to restaurants, sports shops, entertainment and public centers which have the terminals installed. The system displays activities at various resorts, snow conditions and promotional messages for the resorts and for the stores which have the terminals (such as the message shown in the right photo). Skicom, 1281 San Juan Rd., Tustin, CA 92680.



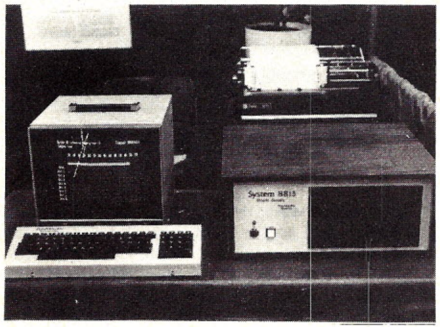
Afraid of the high development costs for a piece of applications software? Well, perhaps you'll get lucky and a class from the Northern Michigan University (at Marquette, MI) will come knocking on your door and volunteer to develop what you need as a class project. That's what happened with a realtor in Marquette! The students developed a 16K TRS-80 version of a program which handles up to 600 listings (listing #, price, location code, style code, status code and date). An updated 32K version will include street and address. Sounds like an ideal marriage between small businesses and the educational community. I hope this gives others some ideas.



NAPIER'S BONES, INVENTED IN 1617, SHOWING THE MULTIPLE OF 2X7=14



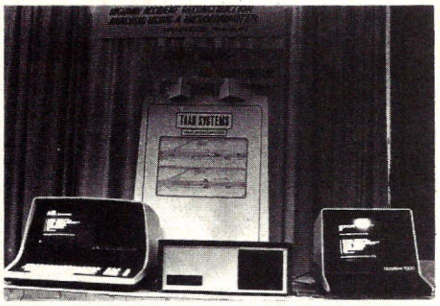
The distinguished gentleman in the photo is Joe Kasser, a member of the NCC Personal Computing Steering Committee and an officer with AMSAT, the Radio Amateur Satellite Corporation. They were demonstrating some exciting things over the horizon (literally) in the years to come with inter-computer communications. The new Phase III satellite from AMSAT, to be launched by NASA shortly, will have provisions for transmitting data to and from computers. This new, worldwide communications medium will be available to anyone willing to invest approximately $1,000 to $1,500 in a (brand new...less for used) transceiver and spend a minimum amount of effort in getting a Technician License. (The Technician License has a code requirement but it's only 5 words per minute...which a 5-year old could easily learn.) For further information on the license and amateur satellites write to the American Radio Relay League, 225 Main St., Newington, CT 06111.



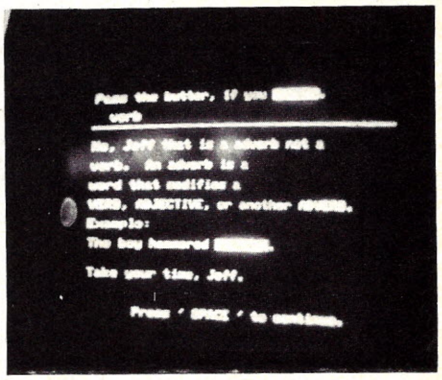
Polymorphic Systems (460 Ward Dr., Santa Barbara, CA 93111) has a comprehensive line-up of business software to go with their 8813 system. And, as you can see from the new desk-mounted system on the right they've combined their 3-mini drive 8813 with two 8" drives for increased capacity. Looks impressive. (One of the software packages they're offering is PLAN, A **P**rogramming **L**anguage for **AN**alysis. This is a business planning/forecasting package which is also available for the Apple II for $95 from Desktop Computers, Box 6791, 5276 Hollister Ave., Santa Barbara, CA 93111. We should have a review of it coming up soon.)

Claudia Nichols, of Advanced Computer Products (Box 17329, Irvine, CA 92713), is showing off their new "No-Name Mainframe." All it needs is a motherboard...and some S-100 boards. Drop 'em a line, along with $1, for a copy of their latest catalog.

NCC, con't...

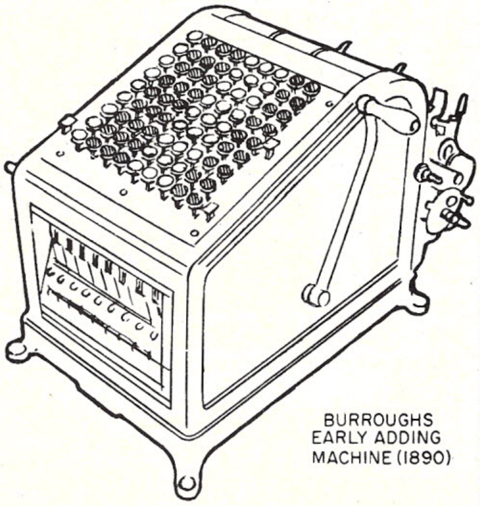


Amit Reizes is a programming analyst. His father, Haim, is a specialist in accident reconstruction and analysis with over 25 years of automotive engineering experience, and the author of a book called **The Mechanics of Vehicle Collision**. The two of them have combined their talents in the development of a program (running under a 32K Northstar Horizon) for the analysis and reconstruction of highway accidents which should be of value to police departments, insurance agents and attorneys. (4616 Glasgow Dr., Rockville, MD 20853.)

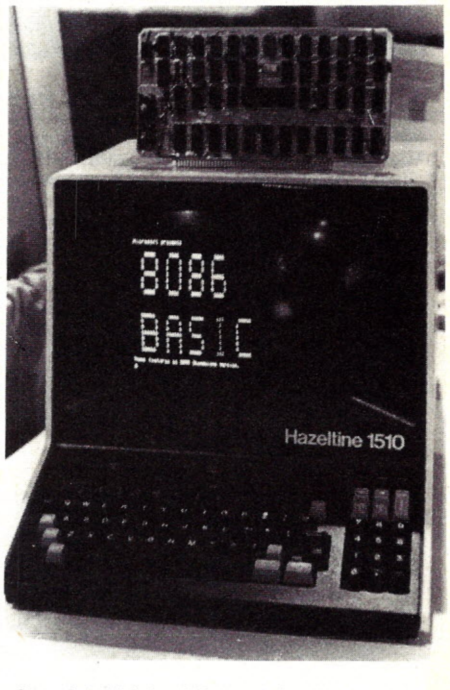


Thomas D. Ankofski (36708 Curtis, Livonia, MI 48152) has developed an interesting home education program (for an 8K PET) for teaching English to 7-year olds to adults. The program has different skill levels and is designed to teach sentence structure and identification in an interesting and comfortable manner (as you can see from the sample lesson at the right). Perhaps we can get Tom to do an article on the program in a future issue of Creative.

Perhaps one of the most significant developments to come out of NCC '79 was this meeting of three brilliant minds (did I say that?) at a Personal Computing VIP Reception held at the famous Stork Club on June 6th. Should we be expecting some new, exotic S-100 board from this trilateral gathering of George Morrow on the left (Thinker Toys), Lee Felsenstein (Golemics, Inc., Berkeley CA...designer of the VDM-1 board and Creative Contributing Editor) and Bill Godbout on the right, owner of Godbout Electronics? Stay tuned...



BURROUGHS EARLY ADDING MACHINE (1890)

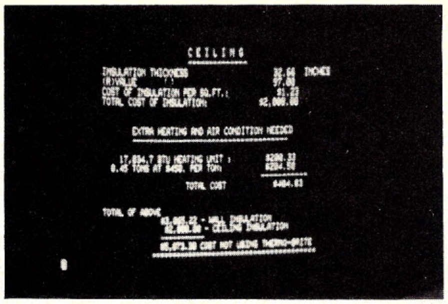


One of the "visitors" in the Lifeboat Associates booth was Seattle Computer Products (16611-111th S.E., Renton, WA 98055) and their 16-bit 8086 S-100 board...running Microsoft 8086 BASIC! The board sells for around $900 and is designed to run with either 8-bit or 16-bit S-100 memory. Lifeboat Associates are still going great guns with new developments. They've recently finished CP/M for the Heath system and are working on a similar version for the Polymorphics 8813. The end result will be three systems (the TRS-80, H8 and 8813) which will all be able to run the same CP/M operating system and software. (Suite 34, 2248 Broadway, New York, NY 10024.)

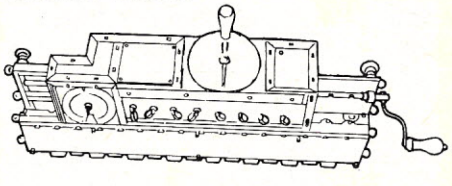
Did I say something about there not being any new hardware at this show? Well, excuuusse me! The Radio Shack Model II is here! My first reaction when I saw those 8" drives was, "Will it run CP/M?" Steve Leininger, Computer Engineering Manager at Tandy and one of the original designers of the TRS-80, replied that he could make it run CP/M without too much effort, but that it wasn't designed with that objective in mind. (Okay, I guess someone else will have to come along and do it.) Needless to say, Radio Shack would prefer you to buy their software and they're putting a lot of effort these days into developing some good business packages (I saw a demonstration of an inventory program that looked quite sophisticated... and they said others were available). You know they're serious about going after the business market when you see that desk with **four** 8" drives!

NCC, con't...

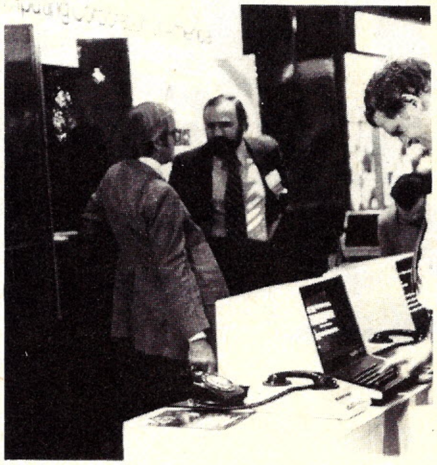
Here's a new TRS-80 to S-100 interface which may, or may not, be an inexpensive way to go (depending on whether or not you already own an S-100 mainframe). The interface board, which plugs into an S-100 slot, includes a parallel printer port, sockets for up to 32K of memory and a floppy disk interface. The board sells for $199 from New England Electronics Co., 679 Highland Ave., Needham, MA 02194 or Micro Computer Devices, 960 E. Orangethorpe, Bldg. F, Anaheim, CA 92801.



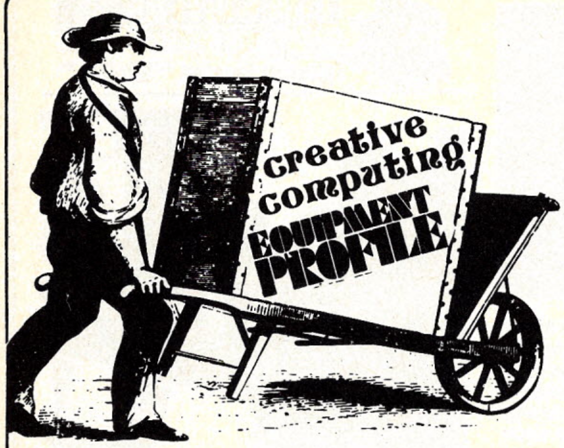
The program on the screen is just one of many routines in a complete energy savings program developed by Nickey Naumovich. The package was developed for helping home, apartment and industrial builders calculate energy-saving steps and savings. The program asks for such things as the type of walls, the amount of space between the walls, wall thickness, the size of the building, types and number of windows and doors and more. Last year Nickey won 2nd Place at the NCC Personal Computing Convention with his program. Since then he's gone on to form his own company and invented a new product for reducing air infiltration: Thermo-Brite. Parsec Energy Research, 9952 Parkford Dr., Dallas, TX 75238.



AN ORIGINAL CALCULATING MACHINE BUILT BY LIEBNITZ IN ABOUT 1694



The folks at Telecomputing Corporation of America have come up with a timesharing system for home and small business users. They have a wide variety of software in home applications, education, games, small business and accounting (general ledger, A/R, A/P, payroll, inventory, simplified data base management, etc.) and language processors. Cost is $2.75 per hour during off-peak hours (6 p.m. to 7 a.m. weekdays and all day Saturday and Sunday). Drop them a line and ask about their system, which they call "The Source." TCA, 1616 Anderson Rd., McLean, VA 22102.



Compucolor II

Steve North

The Compucolor II is a recent entry into the field of consumer-oriented personal computers. It's an improved, up-to-date version of the original Intercolor 8001 microcomputer. The 8001 was first made under the Intercolor name back in the days when Altairs and IMSAIs were all the rage, but Intercolor decided to concentrate on OEMs and business customers and so it formed Compucolor to make personal computers. The 8001 was far ahead of its time, with BASIC in ROM and high-quality color graphics on a large screen, but it was a bit expensive for the personal computer market (at $3000), its "floppy tape" eight-track tape units had chronic compatibility problems and it wasn't S-100 bus standard when that was an important thing. In the past year or two the market for graphics-oriented microcomputers has exploded and Compucolor has had some time to develop a new model. So it was with special interest that we tested the Compucolor II.

Hardware

The computer comes as two units: a keyboard/CPU (central processing unit) and a video display/minifloppy disk drive. The system is based on the 8080 microprocessor, which seems a little odd since there are newer, more powerful microprocessors than the 8080 which also enable less complicated electronic designs. Not really important if you're going to stick to BASIC, though.

The concept behind the design is very good. There's no tangle of cables running from one component to another, just a ribbon cable. Because there's a built-in color monitor, you don't need to monopolize the family TV set for hours, and the display is better. Regular TV's intentionally blur and soften images, but computers don't need compromises that normal broadcasting does. The Compucolor's special video monitor provides much sharper graphics and more brilliant colors.

Compucolor also deserves credit for foregoing audio cassettes in favor of floppy disks. Audio cassettes are plagued with loading problems, slow data transfer rate and can't handle serious data file manipulation. A floppy disk allows you to load programs in a second or two and keep data files on-line for quick access and updates. The tradeoff is that the price of a basic Compucolor II is much higher than that of a cassette-based TRS-80 or Apple because you pay for the disk drive right away. A Compucolor formatted disk has a capacity of 51K, which isn't exactly impressive, though they brag that you can flip the disk over and use the other side. (This does not give you more storage actually on-line because the computer can't access both sides simultaneously, nor do most floppy disk media manufacturers sell single-sided disks for dual-sided applications.) A further minus is that the disks are nonstandard and must be purchased from Compucolor because, ha, ha, you don't get a formatting program with the computer. The concept of not bothering with cassettes at all and including a disk drive with the computer is still good, though the price difference won't help sell any computers. A second disk and a printer can be plugged into the basic unit.

The keyboard comes in three versions. The least expensive (71 key) model is probably adequate for most users, though the other two keyboards have some nice extras such as a numeric keypad, special color and graphics control keys and a BASIC command key so you can enter BASIC keywords quickly. The keyboard is apparently shared with an Intercolor model as some of the legends on the keys have no apparent significance. For instance, the "P" key doubles as an escape function key for "CPU OPSYS." Funny, there's no resident operating system in the Compucolor and escape-P doesn't do anything. There are some other keys which have purposes not yet fathomed by the reviewer.

Most important, the Compucolor II is capable of some pretty spiffy graphics. It has eight colors (black, blue, red, violet, green, light blue, yellow, and white). Points are plotted on a 128 x 128 grid, while the characters are displayed in a 16 x 64 or 32 x 64 format. (There are two character sizes, one twice as big as the other, but both are all caps, a disappointment on a highly graphics-oriented machine. In place of the lower-case characters there are special graphics characters, such as chess men and portions of geometric shapes.) Further, the characters can be set individually to blink. The graphics and characters can be mixed on the display and each block may be programmed for foreground and background color. This allows you to

Compucolor II, con't...

plot green points on a red background or a letter X in black on a violet background. By comparison, the Apple II has 16 colors in low-resolution mode, but plots only on a coarse 40 x 48 grid, and graphics and characters can't be mixed. In high-resolution mode the Apple has denser graphics than the Compucolor, but only four colors (black, white, green, and violet) and text can't be mixed with graphics, either. The Compucolor isn't the ultimate in personal computer graphics but it's one of the best-performing ones we've tested so far.

System Software

The Compucolor II comes with 8K, 16K, or 32K (maximum) of RAM. The rest of the addressable memory is reserved for system programs, including BASIC, FCS (file control system), CRT Mode, and probably some other handy stuff for graphics and I/O.

The BASIC is Microsoft 8K BASIC, with disk I/O and graphics additions by Compucolor. The BASIC is good, but not as good as Microsoft Extended BASIC (a.k.a. TRS-80 Level-II BASIC). The Compucolor II would be a better machine if it did have Extended BASIC, but possibly there was a problem in fitting all the system software and screen maps into just 32K of memory. (A handy argument to remember next time someone knocks 16-bit personal computers with 24 meg of memory addressing space.)

In BASIC, some graphics can be done simply with PRINT statements, because character strings can contain color information. More complex graphics are controlled through the PLOT statement. PLOT is followed by numeric codes (range 0-255) which are apparently just fed to an internal graphics processor. Thus, you program plot codes to control blinking, character sizes, vector plotting and bar graphs, etc. This is a very flexible approach but it's unwieldy since you have to memorize plot codes or refer to a list, and it doesn't make readable programs. Likewise, all the random disk file I/O is done with just three statements: FILE, GET, and PUT. This works, but making one verb do the work of five isn't people-oriented.

Lurking elsewhere in the 32K is a File Control System—a small DOS for listing directories, purging files, copying files from one drive to another and the like. Fortunately, it is possible to get from BASIC to the FCS and back to BASIC without damaging your program in memory. The CRT Mode allows the Compucolor II to be a dumb terminal for hookup to another computer system. The color and graphics are still available by transmitting special codes from the main computer.

An AUTO key on the keyboard automatically loads and runs a menu program from disk, very handy for idiot-proof turnkey operation.

Usually you can get from one program or function to another with only one or two keystrokes. The system powers up in BASIC, and you can return there by typing escape-W. The CPU-Reset key enters the CRT Mode and FCS is entered by typing escape-D. The FCS error messages are typical inscrutable computalk (like EDCS, ENVE, and EWSF). It may be unfair to chide Compucolor for something that IBM has gotten away with for years, but shouldn't personal computers be different? (That reminds me of an ugly rumour going around that a very big microcomputer software house is coming out with a disk operating system inspired by IBM's JCL. Just when you thought it was safe to have a computer...)

Documentation

The documentation that comes with the unit is an 18 page booklet, chock full of at least one or two helpful hints. It's best used as a quick reference for commands, statements, and error messages. The **real** manual (called a Programming Manual) is over 120 pages long and includes helpful information for beginners and experts both. The section on BASIC starts out with variables and gradually explains character strings, arrays, nested loops, graphics, and files. It also includes details about machine-level programming, but probably not enough to get started without some additional information. But you have to purchase this manual separately from the computer. Perhaps Compucolor had in mind that their computer would be bought by users, not programming types.

"Poor dear! Our new puppy chewed up all of the software for his computer..."

Applications

Our Compucolor came with a number of "Sof-Disks," each pre-programmed with several games or applications. In general, the canned programs were well above average in quality. We especially enjoyed Othello (the computer plays a good game), Artillery, Star Trek, Hangman, and Piranha. There are also some useful programs for 1040 forms, computer-assisted instruction and financial planning. The disks cost $20 each. There are over nine disks already, so this is probably the beginning of a good library of software. However, there are very few second-sources of Compucolor application software at this stage.

Overall

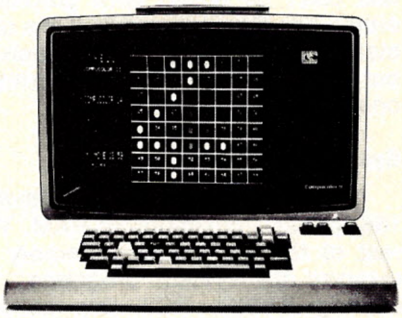
In general, you can't help but admire an idea like the Compucolor II, but the overall impression is that they had a lot of outstanding ideas but didn't quite pull them all together. The BASIC could be improved by better statements for control of graphics and disk I/O. Also, sound-making hardware would be a major plus.

The Compucolor II has a lot of different capabilities and functions and programs inside, but they're not presented to the user in an easy-to-understand manner. I'd have to admit that I think disks and ROMs don't mix. The advantage of ROMs is that they're a cheap way to have instant BASIC. As soon as you add disks, the ROMs tape up valuable space that could be used for RAM and interfere with a well-organized disk operating system. As long as you have disks, you might as well put BASIC on the disks and load it like any other program. However, Compucolor perhaps envisions this as a turnkey machine and didn't want to bother with changing disks or putting BASIC on every Sof-Disk.

The Compucolor could be a really dynamite personal computer if it was redesigned for twice the disk capacity, with 48K of RAM and 16K of

Compucolor II, con't...

ROM or screen maps as needed, to run the CP/M disk operating system. Microsoft BASIC with graphics extensions and other useful system software could be kept on disk. A system like that would easily be worth twice the present price of the Compucolor, though it would be out of the league of inexpensive personal computers.



In brief, the Compucolor II is worth your consideration if you're especially interested in high-quality color graphics, and don't need the reassurance of all the TRS-80 owners on your block to know you bought the computer that's right for you. ☐

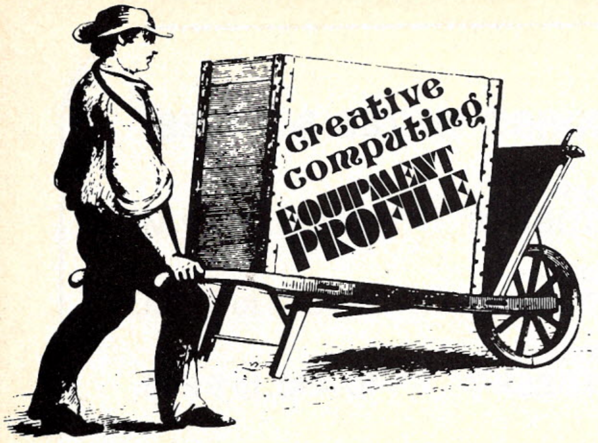
Next month we'll have another evaluation of the Compucolor from a person who has owned it for several months and learned to really use its capabilities.

Another view...

We have had several young children learning to program in BASIC on the Compucolor. Our observations indicate that while in most ways a beautiful system, the Compucolor has several idiosyncrasies that may prove to be annoying. For example, there is no SCRatch or NEW command to erase a Basic program in memory. Most programming texts use this extensively to erase an existing line or short program before going on to the next one; with the Compucolor one must hit Control/Shift/CPU Reset and wait about 5 seconds for the machine to restart itself.

Also, the absense of a built-in speaker for music or sound effects is a bit surprising on a system in this price range.

Nevertheless, the kids love the spectacular color graphics and, eventually, have learned to adjust to the above idiosyncrasies. —DHA

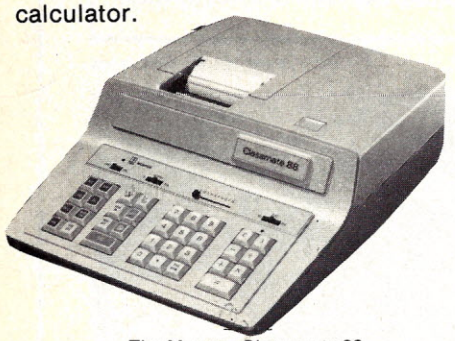


Monroe Classmate 88 For The Classroom

Burchenal Green

Teaching the Fundamental Skills

The ability to master basic skills as a foundation from which learning can take place is important. For example, a student should be able to add, subtract, multiply and divide numbers. He can then figure problems using these skills. If he then uses a calculator to do the computation quickly, freeing his time to solve more complex problems, it is probably of educational advantage. But first he must know the basics of math to be able to properly utilize the calculator.


The Monroe Classmate 88.

Few educators would argue that the learning of fundamental skills is a prerequisite to education. Therefore, it is the responsibility of educational institutions to evaluate student's skill abilities, then facilitate the most effective means possible of helping students learn. Unfortunately, it isn't always true that basic skills are acquired in the early grades. At every grade level, including college, there is a percentage of students who can't do simple math. So, regardless of present grade level, student's skills must be diagnosed, they must be taught the concepts needed and the practice provided to help them master the skills lacked.

Technological advances seen in the last decade should have been used on this serious problem. Had education worked on developing resources to meet its needs, every student in this country ten years ago would have had access to computers and innovative, interactive programs designed to individually tutor math, music, art language, problem solving, etc. But educational institutions on the whole are not at the forefront of knowledge utilizing technological advantages to help students learn. Nor do they command or properly allocate necessary funding. So only now are computers beginning to creep into the classroom in any significant numbers.

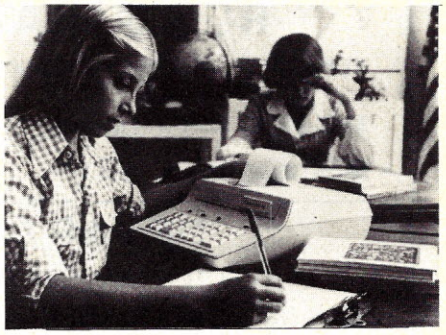
Operation Achievement

The Monroe Classmate 88 is a welcome exception to the technological lag. It is a calculator containing seventy hardwired programs that provide drill and practice problems in addition, subtraction, multiplication, division, fractions, decimals and in number concepts. Obviously designed with the classroom in mind, the Classmate 88 is an attractive bright yellow and beige with some green and orange keys, is durable but light weight, and is easy to learn to use. The machine is well thought out and can be utilized to provide individualized drill and practice on many skill levels in basic mathematic fundamentals. It can be used on all grade levels where students need help in mastering basic math skills.

Although the calculator abilities of the Classmate 88 are certainly useful, it is the Operation Achievement mode that makes it indispensible to the classroom.

Monroe's intention is not that this calculator be a teaching machine, but will reinforce the concepts the teacher has already instilled, with drills on a one to one basis.

The necessary support material for the Classmate 88 is well organized and easy to use. It is recommended that a student be limited to fifteen minutes of drill and practice a day, and that the drill be in a skill already taught to the student. Of course after a lesson in multiplication, for example, a teacher could assign a student the Classmate 88 to practice problems at her current level. But since children's skill levels are so diverse, diagnostic materials are provided by Monroe. A survey test aids the teacher in determining which placement test the student should take. The placement tests have a fixed scoring pattern (for which the teacher uses the Classmate 88 to computate the grade) to ascertain which of the many skill programs the student needs first. This starts the student at her own level.


Math is more fun when a student can practice on the Monroe Classmate 88.

Once the teacher has assigned a skill level for a student, almost anyone can learn to operate the calculator. A book of flowcharts provides the easy step by step procedures to be followed. The flowchart in Figure 1 is a sample of the ease which the student can work the Classmate 88. It is important to a teacher that the students can operate the machine unassisted, freeing the teacher to work with other students.

The turn of a switch determines the number of problems to be worked on as 10,25 or infinite. If the teacher or student chooses an infinite number of problems, the student need only press the SCORE key to end the program. The date and student number will be recorded in red on the tape by using the DATE key. Then, following the flowchart, the student will key in the program she wants to work on. If

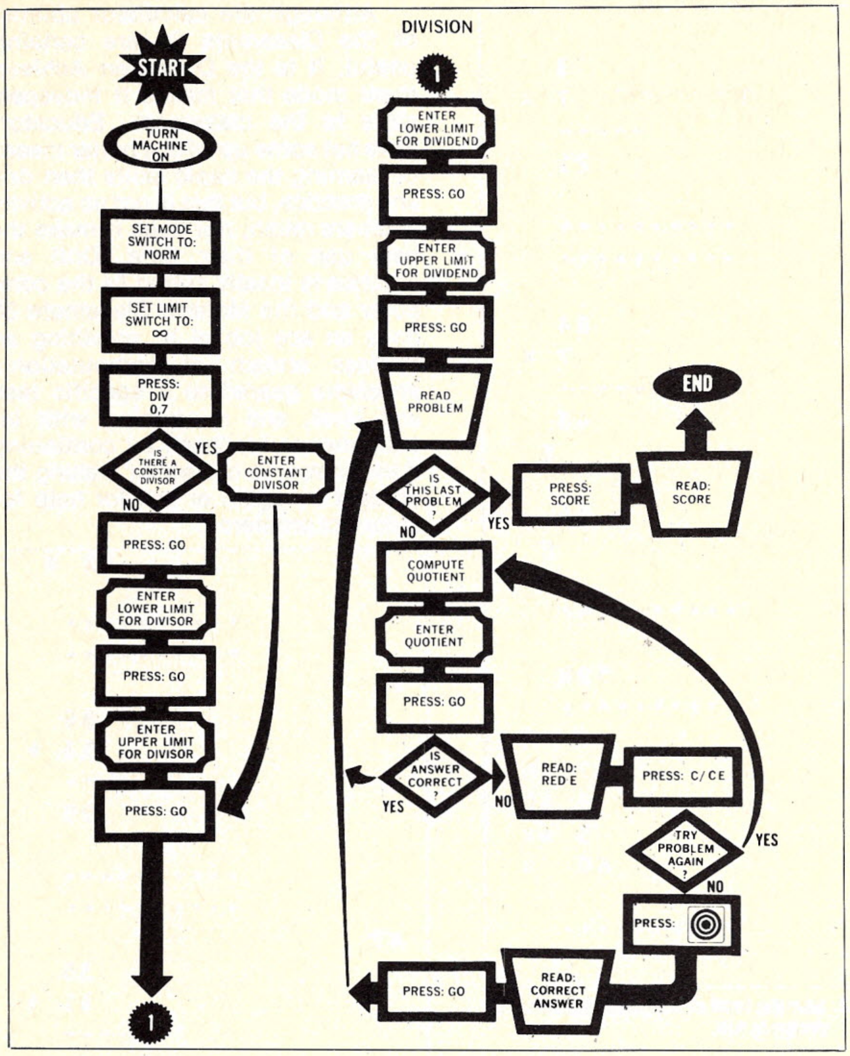


Figure 1. Flowchart of procedures to follow to run the Multiplication of 07 program.

she keys in ADD 0 3, for example, an addition problem will be printed using 2 digits and 2 digits with regrouping in units. The student must then answer the problem appearing on the tape. If the problem is 15 + 12 she would touch the 2 and 7 keys and then the GO key, which enters her answer. Here, of course, it would be nice for the student to be able to figure the answer on the calculator. That means the 7 key would be touched first since the student would be adding the 5 + 2, or units column, first:

$$\begin{array}{r} 15 \\ +12 \\ \hline 7 \end{array}$$

Monroe did consider this option when designing the Classmate 88, but decided there were too many different algorithms used in computation, and so designed the calculator to respond only to the correct answer. Students are encouraged to work with pencil and paper before keying in their answers.

If the student records the correct answer, the calculator responds with a double row of dotted lines and another problem in the same skill area. If she gets the wrong answer, a red E for error appears on the tape. She may try as many times as she wants to get the correct answer, but will get a red E after each incorrect answer. When she wants help (but only after an initial try at the problem), the student pushes ◎ and the correct answer and a red dotted line appear. When she has finished all the problems, she touches SCORE and is given the number and percentage of correct answers.

The student's tape then becomes a diagnostic tool for the teacher. A red E appears beside all problems missed and a dotted red line is printed under any answer the machine was asked to provide. The teacher can instantly determine trouble spots for further work, and the use of tape as hard copy is a plus for the Classmate 88.

Figure 2 shows a sample tape for a student following the flowchart. Multiplication review problems with limits and a constant self-determined would be provided. The code for the program is on the tape "07M." The lower limit is "1", "100" is the highest limit and "7" is the constant. She is then given a problem which she answers correctly. The second problem is missed and gets a red E next to the try. She asks for help and gets the

Any school without a Classmate 88 is doing a disservice to its students.

dotted red line indicating the following answer was provided by the machine. She answers numbers 3 and 4 correctly, but misses the answer to #5 twice before she gets help from the machine. She then asks for the score and discovers that in Program 07 M she got 3 correct answers out of 5 answers for a score of 60%. It is clear that she still needs some work in multiplication with 7s.

In Figure 3 the student wants to add 2 digits and 2 digits with no regrouping. No limits are needed, so the student need only press the program code ADD 0 2 keys. Here she gets the first two problems correct, missing the third problem twice before she asks for the answer, then her score.

The Classmate 88 contains six programs for which the teacher or student can designate the upper and lower limits of numbers to appear in the problems given and can select a constant. If red marks on the tape show that a student has trouble with sevens, all the problems in the student's next program can deal with the seven. A handy feature.

When the Classmate 88 is used in this fashion, as a drill and practice device, it is in the Operation Achievement mode. The seventy programs possible in Operation Achievement are as diverse as Addition of 2 digits plus 2 digits with no regrouping; Subtraction of 3 digits minus 3 digits with regrouping in units and 10's; multiplication of 1 digit x 1 digit x 1 digit; Division Review with remainders; limits on divisor and dividend with constant divisor option at the user's input; Equivalent Fractions; and Fraction to Decimal Conversion rounded to four decimal places. Besides the skill levels in Addition, Subtraction, Multiplication, Division, Fractions and Decimals there are seventeen wonderful programs entitled Number Concepts. These run the gamut from giving practice in

```
        07   M
         1
       100
         7
..........
..........
#1
        33
         7 x
     -----
       231
..........
..........
#2
        34
         7 x
     -----
         1
           E
..........

       238
..........
..........
#3
        32
         7 x
     -----
       224
..........
..........
```

```
#4
         3
         7 x
     -----
        21

..........
..........
#5
        84
         7 x
     -----
        48
           E

        21
           E

..........

       588
..........

        07   M
         3  AC
         5  A*
        60   %

..........
```

Figure 2. Sample tape of a Multiplication of 07 program run.

factoring a number to its prime factor, to presenting number puzzles (always an attraction), to offering drills in estimating multiplication answers. Figure 4 shows a tape giving a few problems in Number Concepts: 11, Addition Cross Number Puzzle; 14, What's My rule; and 16, Multiplication With Estimation.

The Classmate 88 can be used two other ways in the Operation Achievement Mode. If a teacher wants to create homework problems, he need only set a switch to P and a series of problems without the answers in the skill area he designates will be generated. If he wants a set of problems with answers for a test he needn't bother with the time consuming task of making up problems and figuring out the correct answers. He has only to push the P/A switch and he'll get a set of problems with answers. Monroe is quick to point out that, run as P/A, the problems are not scored and the ambitious youngster who tries to run this instead of working out his problems will get a score of "0". These are two excellent time saving features that extend the capabilities of the Classmate 88 to be more than a drill and practice tutor.

As a Calculator

In the Calculator mode the Classmate 88 has the ability to do not only the mixed operations of addition, subtraction, multiplication and division, but work problems with three open parentheses at any time. The tape for problem

$$3 + \left[\frac{28}{15-(4x2)}\right]$$

appears in Figure 5. This is a great feature for the classroom.

It also can convert a fraction to its decimal equivalent by the touch of NUM/DEM as the tape in Figure 6 demonstrates for 3/4, 7/8 and 14½.

As with many calculators, the Classmate 88 is able to use the first number in a multiplication or division problem as a constant factor, which is a help in practicing those time honored times tables. Figure 7 demonstrates the few keys needed to do multiplication and division with 7 as a factor.

Indespensible Tool

Although the calculator abilities of the Classmate 88 are certainly useful, it is the Operation Achievement mode that makes it indespensible to the classroom. Educators have not come up with a better means of learning the basic skills than drill and practice, but this is not an activity teachers relish, nor does it make the best use of their time. Drill and practice is ideally suited to the computer and the Monroe Classmate 88 does an ace job of it, providing an endless stream of computational problems geared to a specific task and level, and reinforcing with instantaneous feedback by continuing if the answer is correct or making the student try again or ask for help for incorrect problems.

```
          02  A

..........
..........
#1
          86
          13 +
        ----
          99

..........
..........
#2
          35
          41 +
        ----
          76

..........
..........
#3
          69
          30 +
        ----
          39
             E

          21
             E

..........

          99
..........

          02  A
           2  AC
           3  A*
          66   %
..........
```

Figure 3. Sample tape using the Addition 02 program.

```
          11 NC

    ..........
    ..........
#1
    6  4  7

    9  7 10

       4  7

             6
    ..........
    ..........
#2
   12   8 10

   17  13 15

   11   7

             9
    ..........
    ..........
#3
   13  13 16

    6   6

   12  12 15

    ..........

          11 NC
           2 AC
           2 A*
         100  %

    ..........

          14 NC

    ..........
    ..........
#1
             9
             2
           ---
            18

             4
             7
           ---
            28
```

```
             3
             3
           ---
             9

             3 M
    ..........
    ..........
#2
             9
             3
           ---
            12

             2
             1
           ---
             3

             4
             8
           ---
            12

    ..........
           14 NC
            1 AC
            1 A*
          100  %
    ..........

           16 NC

    ..........
    ..........
#1
           93
           52 x
         ----

          500
              E

         5000
```

```
      ..........
      ..........
#2
             699
             182 x
            ----

             100
               E

      ..........

          127218
      ..........

             16 NC
              0 AC
              2 A*
              0  %

      ..........
```

Figure 4. Sample tape of a few problems in Number Concepts

The Classmate 88 can't branch to difficulty levels depending on the student's response, and it isn't tutorial. These reasons could be cited for not getting a Classmate 88, instead dreaming of Plato, Turtle or the Dynabook in the hands of Everychild. It's a nice dream, but this isn't tomorrow. And while microcomputers are beginning to enter schools with wonderful CAI capability and potential, none of them can now do what the Classmate 88 not only does but has done tried and tested from 1976. At $745 it is not an inexpensive machine, but definitely affordable. Any school without a Classmate 88 is doing a disservice to its students.

When a Classmate 88 is delivered, by a Monroe representative who conducts a workshop describing its

```
 3•  +
 3•  (
28•  ÷
28•  (
15•  -
15•  (
 4•  x
 2•  )
 8•  )
 7•  )
 4•  =
 7•  *
```

Figure 5. Tape of the problem $3+\left[\frac{28}{15-(4x2)}\right]$

Monroe, con't...

```
  0•3    /
   4•    =
 0•75    *

  0•7    /
   8•    =
0•875    *

 14•1    /
   2•    =
 14•5    *
```

Figure 6. Tape of Converting 3/4, 7/8, and 14½ to their decimal equivalents.

features and tips on successful utilization. It is also possible for schools to modify the machine for use with handicapped students. For more information on the Classmate 88 contact Peter S. Cybuck, Educational Consultant, Monroe, The Calculator Company, The American Road, Morris Plains, NJ 07950. □

The Classmate 88 Gets A Report Card

Monroe Calculator Company is a neighbor of Creative Computing in Morristown, NJ so we took advantage of their good neighbor policy to borrow a Classmate 88 for a semester. One of the local teachers utilized it with great success with her elementary school class and provided the following evaluation.

The students in my class were able to use the Monroe Calculator 88 independently, either alone or in small groups in our learning center, which made it an invaluable aid to me as a teacher.

All students were eager to use the Classmate 88 when it was first introduced to them and this enthusiasm continued even after it became a more familiar part of the classroom.

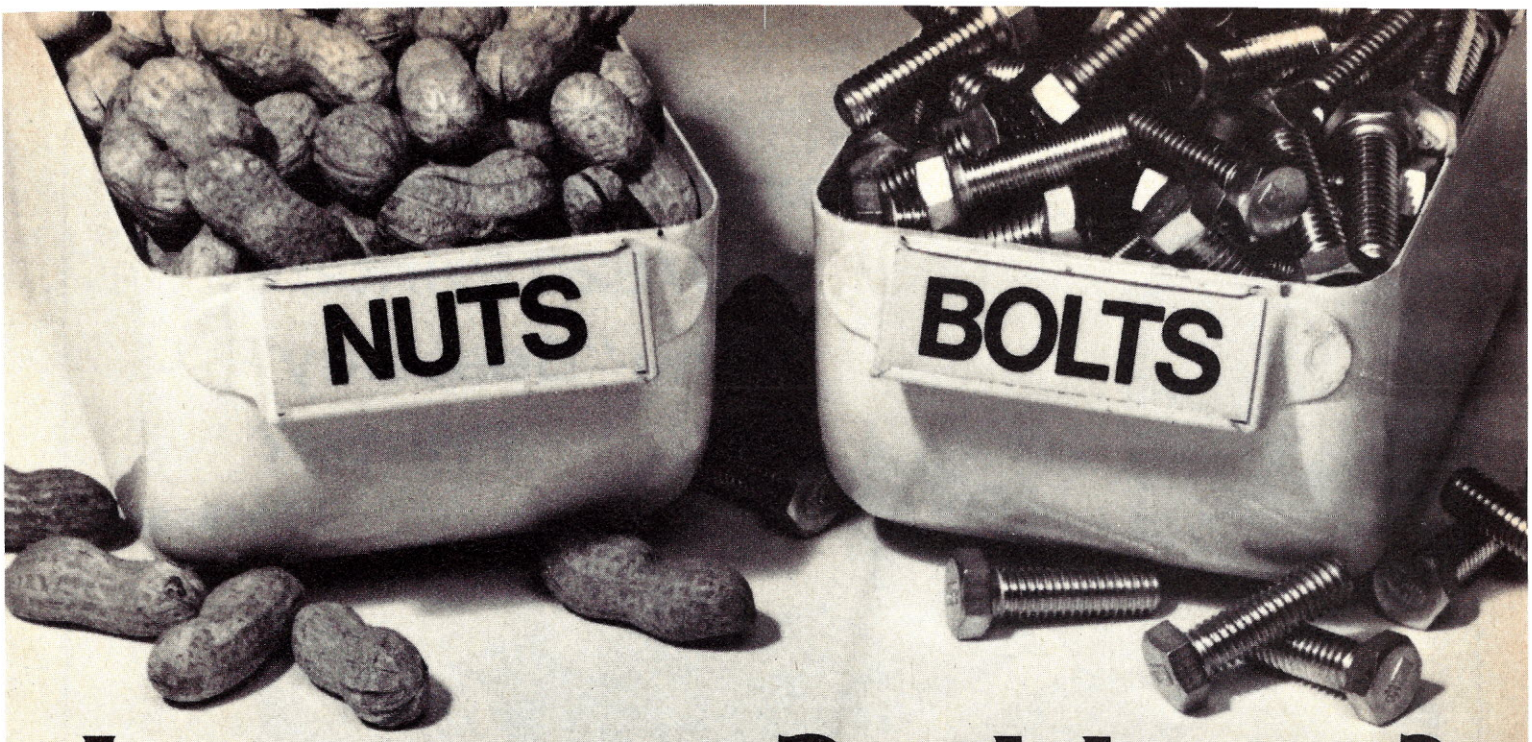
Once children had the opportunity to learn how to operate the Classmate 88, I found it interesting to listen to their comments from which I picked up valuable clues for further worthwhile assignments, based on the child's interest and/or needs.

It appears that the Classmate 88 helps develop a positive attitude on the part of children towards independent work in mathematics.

Rita Roth, Teacher
Alfred Vail School
Morris School District, NJ

```
 7•   ×
 5•   =
35•   *

 6•   =
42•   *

 7•   =
49•   *

 8•   =
56•   *


56•   ÷
 7•   =
 8•   *

42•   =
 6•   *

35•   =
 5•   *
```

Figure 7. Using 7 as a factor in multiplication and division, the tape shows how few keys need be pushed.



Inventory Problems?

Are you having trouble keeping the right nuts and bolts in stock? Since even a simple mistake can cost you time and money, a good inventory system should do more than just count parts. It should tell you exactly what you need, when you need it, where to get it, and how much it will cost.

The MSI Inventory System Seven enables you to maintain a versatile data base for controlling inventory. It lists part number, description, quantity on hand, vendor, cost, selling price, optional pricing, usage levels for previous month, present month, and year-to-date, and much more.

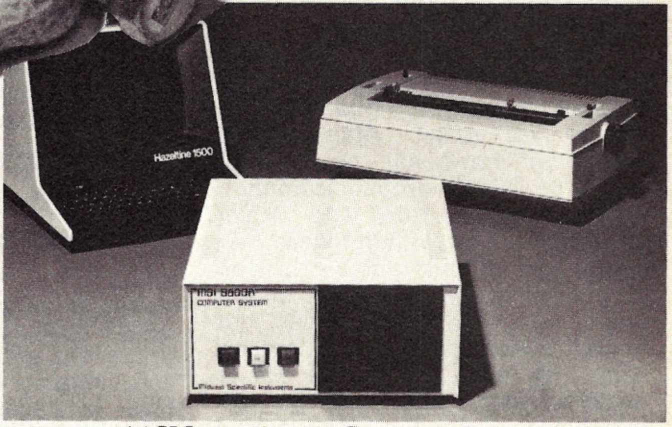
When quantity on hand items reach minimum levels, the System Seven compiles an automatic reorder list. This list can be generated by specific vendor as well as a complete listing of all materials to be ordered.

In addition to the item listing, the Inventory System Seven "bill of materials" provides you with a complete inventory of items used in the manufacture of subassemblies and complete products. It also contains other cost items such as labor costs, total raw materials costs, and miscellaneous costs.

The MSI Inventory System Seven is built around the versatile MSI 6800A Computer with 56K of RAM. An integral dual mini-floppy memory gives you an additional 630K of memory and makes inventory control fast and efficient. The System Seven will interface with any industry standard CRT, and you have the option of both a "daisy wheel" word processor for high quality document preparation and a dot matrix printer for high speed production.

The System Seven can be expanded to handle all your data processing needs or you can select one of nine other MSI systems now available for business, industrial, scientific, educational, and personal applications.

If you need more than just a nuts and bolts inventory system, we have more information about how the Inventory System Seven can solve your problems economically.



MSI Inventory System Seven

MSI

Midwest Scientific

220 W. Cedar, Olathe, Kansas 66061, (913) 764-3273
TWX 910 749 6403 (MSI OLAT), TELEX 42525 (MSI A OLAT)

CIRCLE 167 ON READER SERVICE CARD

Texas Instrument's 810/820 Printers: The "Best Buy" Printer?

J. Tom Badgett

The TI-810 in "operating position." It sits atop a homemade desk which also houses my computer and disk drive. That's a Micro-Term ACT-IVB to the left of the printer.

Interesting comments on TI's 810 and 820 printers...some which contradict "official" statements from TI.

One of the first things a computerist learns when he starts programming is that hard copy is all but a necessity for program editing and structuring. The second thing he learns is that he has to pay really big bucks for a printer with reasonable speed, versatility and good looking print. Because of its mechanical nature, the computer printer still is one of the most expensive components of any computer system. Figure on spending anywhere from 1K to 3K dollars for a decent unit.

Even given these price considerations, however, there are some good buys and some not so good buys on the market today. After several months of travel and hours upon hours of hiking around computer shows, terminal fairs and dealer showrooms, I have found what may be the best computer printer buy on the market today: the Texas Instruments' Model 810/820 series dot matrix printer. The 810 is a Receive Only (RO) model while the 820 is designed as a communications terminal and contains a keyboard, a larger character buffer and answer-back circuitry. The printing mechanisms of the two printers are identical, though the available options aren't the same. The biggest difference seems to be that the 820 is not available with a parallel interface.

J. Tom Badgett, 400 Albemarle St., Bluefield, WV 24701.

Features and Options

For most microcomputer applications the 810 is probably the logical choice. The basic printer with upper case only is $1,895.00 retail, though a number of suppliers who advertise in the personal computing and trade magazines discount this basic printer several hundred dollars. That price might be enough to make some

The standard features and available options for the TI-810 are also very impressive.

people gasp, but keep in mind the printer is complete with tractor feed, 256 character buffer, RS-232 interface, intelligent bidirectional print mechanism and 150 character per second printing speed. Compare those features to other printers generally offered to small system users, such as the Centronics 779, and the price is extremely competitive. With tractor feed and parallel interface the Centronics printer is usually priced around $1,695.00.

The standard features and available options for the TI-810 are also very impressive. The basic printer is set up for 132 columns, but has a fully adjustable tractor mechanism which will take narrow forms (such as mailing labels) down to about 3 inches. It contains a self-test feature which will print the standard character set in an offset "barber pole" pattern. That is, it prints the entire

A set of switches on the front panel put the printer on or off-line, align the paper (up or down in small increments), reset an error condition, cause a line feed or advance to the top of the form. Additionally there is a set of DIP switches under the front cover which select the baud rate (from 110 to 9,600 Baud in the standard configuration), parity checking, automatic line feed override and automatic perforation skip override. (NOTE: when using these programming switches you have to turn off the power momentarily, then back on after the switches are set to effect the new programmed configuration.)

Each standard printer also has a paper-out switch which works in conjunction with a warning light on the front panel. There is also a programmable forms length resident in the standard printer. The default forms length is 66 lines, but under software control you may specify anything from 3 to 112 lines. Also standard is software or hardware switchable line spacing, either six lines per inch or eight lines per inch.

You don't have to pay extra for software control, either. By including in your software certain control characters, you may take the printer off-line or put it on-line, do horizontal and vertical tabs, advance to top of form, select number of columns to be printed and choose other operating parameters.

Interfacing

As previously mentioned, the RS-232 interface comes standard. It includes switchable baud rates to 9600 and standard handshaking signals. A number of interface options are available. The one I use is what TI used to call the "Centronics Compatible" interface. A factory engineer tells me they no longer may use that term, and, indeed, assured me that the parallel interface was no longer compatible with the standard Centronics interface. Such is not the case, however, as the parallel card in my OSI Challenger was designed for the Centronics printer and it works well with the TI-810 printer. As further proof, I was telling a Radio Shack dealer recently about the model 810 and he became very interested. I placed the parallel Model 810 next to his Radio Shack and removed the parallel cable from his Model 779 printer and plugged it into the TI machine. The only change required was to turn on the automatic line feed feature as the Radio Shack printer software doesn't send an automatic line feed with the carriage return.

When considering a parallel interface for the Model 810 one must be sure to specify the proper interface, as there are a couple of choices. A dealer in California told me that to get a Parallel Interface compatible with the Centronics Interface I would have to buy the TI LBP (Line Buffered Parallel Interface), which is a $150.00 option. I called the factory and was told the same thing by a company engineer. Further research showed, however, that TI's PLL (Parallel Interface with Character Buffering) has all the required signals for a Centronics-type interface, and costs only an additional $45.00. This is the interface I am currently using with my Challenger II at 9600 Baud and it works flawlessly. The RS-232 Interface comes on the 810 even if you specify current loop or parallel interface options.

Be careful when ordering a TI-810. A few national distributors are ordering the printer in large quantities and are offering attractive discounts. These printers are being supplied in standard configurations only: RS-232, upper case. If you want Parallel Interface it'll cost $150.00 even for the character buffered option for installation outside the factory. Some options, such as the forms controls, may not be installed in the field. Make sure you're getting a printer configured the way you want to use it before you buy at a discount.

Also available are RS-232 interfaces with line buffering and Current Loop with character or line buffering. Other options include international character sets, programmable and switchable forms length, compressed and expanded print, Lower Case, tear bar, paper catcher, stand, etc.

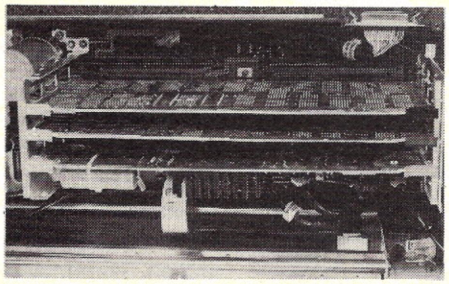
Note that the standard TI-810 uses a high quality 9 x 7 matrix. The company says it produces excellent results on up to six copies, though by the sixth copy the letters are somewhat smeared. Up to four copies (an original and three carbons) are quite good, however. With a parallel interface the 810 can print a continuous

This printing is an example of 10 Charac-
ters per inch horizontally and 6 lines
per inch vertically. The printing can be
compressed both horizontally and verti-
cally.

This printing is an example of 10 Charac-
ters per inch horizontally and 8 lines
per inch vertically. The printing can be
compressed both horizontally and verti-
cally.

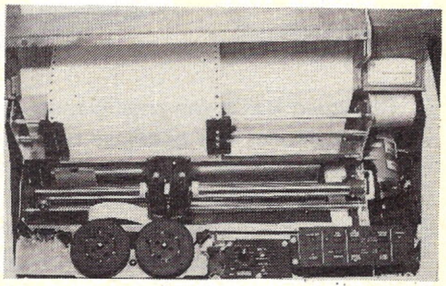
This printing is an example of 16.5 Charac-
ters per inch horizontally and 8 lines
per inch vertically. The printing can be
compressed both horizontally and verti-
cally.

This printing is an example of 16.5 Charac-
ters per inch horizontally and 6 lines
per inch vertically. The printing can be
compressed both horizontally and verti-
cally.



The printer electronics are housed in an enclosed cage at the rear. The cards are what I'd call military-type construction: heavy lands, solder masked and coated with a heavy protective layer. The top of the card cage is padded and fits firmly against the tops of the circuit boards, holding them in place.

Parallel and RS-232 circuitry is included in the unit shown. For TTY interface another circuit board is plugged into a socket below the present PLL connector.



An Inside view of the TI-810 printer. The lever on the right of the print head rail adjusts head clearance for various forms thicknesses. The DIP switches, forms length switches and print compression controls are visible in the lower center of the picture. Front panel control switches are on the right.

810/820 Printers, con't...

150 characters per second. That's 60 full 132 character lines in a minute, or up to 440 lines per minute if the lines are 10 characters or less. You can operate the printer with the RS-232 interface and no handshaking at a continuous 1200 Baud, slightly reducing its printing speed.

The printer is a joy to use. The software programming features provide for easy design of printer output.

The printer is a joy to use. The software programming features provide for easy design of printer output. The 810 is quiet (below 60 db, the company says, with the supplied plastic sound baffle). It operates noticeably quieter than the Centronics 779 and it is worlds away from my ASR 33 Teletype in noise production. One reason the 810 has such good printing speed is the fact that the print head tabs to the next printing position rather than moving at the slower standard printing rate. If you're printing full lines the head seems to sweep back and forth, printing in both directions in a smooth, continuous action. If, however, you are printing a chart or other columnar information, the head leaps to the next printing position. It is interesting to watch the head try to decide whether to move right-to-left or left-to-right.

Service and Reliability

I've only used the printer a short time at this writing, but company supplied figures on reliability and time to repair look promising. Texas Instrument says the Mean Time Between Failure (MTBF) of the entire printer, excluding the printhead, is 2,000 hours, "regardless of duty cycle." The printhead, says Texas Instruments, can produce 150 million characters, while the ribbon is good for up to 7 million characters. Average time to repair the printer, including the print head (which is merely replaced in the field) is 30 minutes. Several service options are available. Some TI dealers pick up service themselves, either through service contracts or on-site service at an hourly rate. Texas Instruments has service centers at strategic spots throughout the country and sells annual maintenance contracts for about 16% of the retail cost of the printer. Also available is depot maintenance, where you pay a lower service rate and are required to ship the printer to TI for in-house service in the event of difficulty. This service won't cost you anything additional beyond shipping. TI will also service your printer on an hourly basis. Presumably circuit cards and other repair parts could be purchased from the company should an end user wish to attempt his own service using the maintenance manual provided with the unit.

The construction of the Model 810 is rugged, compact and high quality. As the accompanying photographs show, the electronics components are housed in a completely enclosed card cage, cooled by a muffin fan. The top of the enclosure is padded with foam and presses firmly against the top of the circuit cards, ensuring that they will remain in place even under conditions of vibration or when moving the printer to a different location. The print head rides on a permanently lubricated stell rod on which it floats freely.

When I first began investigating

the TI-810 printer in the summer of 1978, a Washington, D.C. sales rep told me delivery times were running a nominal 290 days! I gave up on ever getting one. Texas Instruments now has a plant dedicated to the 810-820 printers and delivery times are a more realistic 30-45 days. That's not bad when you consider each printer is built on order. There's no such thing as ordering a printer from TI "off the shelf." Because of the wide variety of options, each printer is built after an order is placed for it.

Documentation and Miscellaneous

The documentation with the printer is excellent. After struggling through a variety of service and operator's manuals produced by several microcomputer companies, it's nice to find a company with the background and personnel to provide understandable manuals with their equipment. The Model 810 has been around for quite awhile, so they've had time to come up with acceptable documentation. A fairly complete service manual is also available for $15.00 which includes, among other things, a quick reference card on software commands.

Nothing is perfect, of course, but my complaints so far on the TI-810 printer are few. The most glaring difficulty is the three-way power plug where it attaches to the printer. For some reason it does not fit snugly into the printer jack and it is easy to wiggle the power plug loose when you turn the printer on (the on/off switch is located adjacent to the power connection). The power plug sometimes falls out when changing paper. I've never had the power connection disengage while the printer was operating, however. The only other problem is just a minor irritation. The paper out switch is located just below the metal platen for the printhead on the extreme left side. If you're not very careful when changing paper the perforations will snag on the switch, either tearing the holes or wrinkling the paper so you have to tear off a sheet and try again.

Otherwise, I have no complaints. The printer works as advertised and works very well. You'll be hard pressed to find more printer for your money anywhere else. □

Texas Instruments' 810 & 820 Printer Specifications

PRINTER

Microprocessor controlled, bi-directional print head. Prints 64 Limited ASCII Characters (810), standard. The Model 820 comes standard to print 95 ASCII Characters. 9 x 7 Dot Pattern.

SPEED: 150 Characters per second

LINE SPACING: 6 Lines per inch (810 has selectable 8 LPI).

HORIZONTAL SPACING: 10 Characters per Inch. (16.5 CPI Optional)

PAPER FEED: Rear or Bottom Feed. Includes Self Test Feature.

COMMUNICATIONS

Serial EIA RS-232C. A standard 25 pin connector is mounted on the rear of the printer.

SPEED: Selectable baud rates of 110-9600 with handshaking.

PARITY: Check for Odd, Even or None. (820 transmits Odd, Even, Mark or Space).

RECEIVE BUFFER: 810 256 Characters.
820 640 Characters.

MODES: Half Duplex, Full Duplex, Half Duplex with Reverse Channel.

MISCELLANEOUS

FORM WIDTH: 3 to 15 inches.

KEYBOARD (820 Only): Full ASCII, Typewriter type with N-Key Rollover. Terminal and Communications Status indicators.

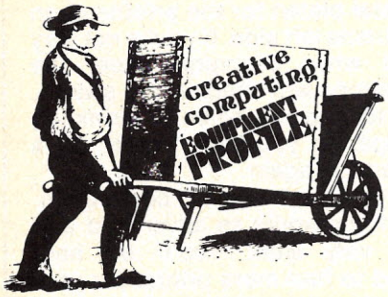
PAPER TYPE: Continuous feed, fanfold or multi-part (6 Parts)

POWER REQUIREMENTS: 90-130 VAC, 47-63 Hz., Single Phase.
180-260 Vac, 47-63 Hz., Single Phase.

PHYSICAL: 810 25.75 x 8 x 20 Inches, 55 Pounds.
820 26 x 8.25 x 21 Inches, 40 Pounds.

OPERATING ENVIRONMENT: 5-40 Degrees C, 5-90 Percent Humidity.

NOISE: Less than 60 dB 3 feet from front of unit.

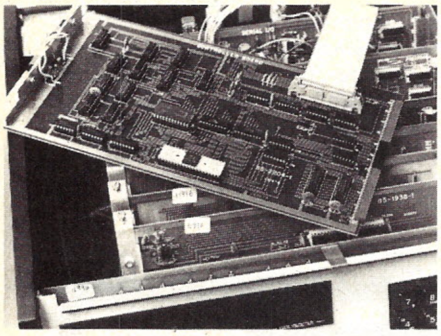


From Benton Harbor:

The Heath H17 Disk System

Donald E. Skiff

The Heath H8 microcomputer has been around for a couple years now, and while people predicted a rush of compatible peripherals, it hasn't exactly worked out that way. Godbout Electronics offers a 12K memory board (inexpensive), and Info 2000 advertised a Z80 CPU board and 8" floppy disk system (very expensive) for it. Beyond the Info 2000 package and the usual games, there hasn't been much non-Heath software yet, either*. So those of us who "went Heath" have been eagerly waiting for bells and whistles to come out of Benton Harbor. Components that don't have to interface to the unique Heath bus, such as printers and video terminals, can be chosen from a selection; for others, such as 5" floppy disk systems, we've had to wait. The disk has arrived.



The H-17 Controller Board: with 2K bytes ROM and 1K bytes RAM (which is write-protected); all bus lines buffered; disks have 40 tracks 102.4K bytes each.

The H17 uses a Wangco Model 82, 5" drive, with 10 hard sectors, single sided and single density. The drive and its circuit board come assembled, even if the system is purchased as a kit. The cabinet, power supply and controller board are by Heath, and can be bought assembled or as a kit. The cabinet accommodates two disk drives, and a plate is furnished to cover one opening of a single drive system. I purchased a kit with a single drive, then added a second drive shortly after.

Donald E. Skiff, 2448 Vera Ave., Cincinnati, OH 45237.

Construction

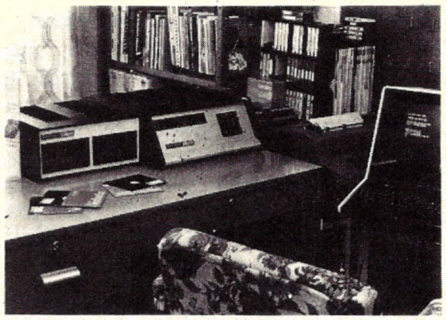
The single-drive system took about 12 hours to build. With some previous experience, I encountered nothing difficult, except for some minor parts not being in the container they were supposed to be in, but there were no missing parts. Assuming that the builder has already assembled the H8 computer, this kit is pretty simple.

Making it run was, for me, another matter. I bought the drive and controller board alone, anticipating a week or so of construction before I was ready to try it out. I didn't have any software for it, and the construction and operation manuals gave no information about actual operation beyond some preliminary tests.

The first test is a memory test, for both the computer and the controller board (the board has 2K bytes of ROM and 1K bytes of write-protected RAM). Then the drive control circuits are checked through the H8 front panel. Each drive is given a read/write test next, and if that fails (mine did on one drive) the book provides a 69-byte speed display program to be entered manually into the computer. A speed adjustment on the drive can be turned (a very sensitive adjustment) to get the drive up to specifications. The manual procedure at this point was unclear, and I went through the process a half dozen times before I realized what I was supposed to be doing and looking for. I finally got the drive to produce the necessary display, and I turned it off to wait for the arrival of my operating system.

Meanwhile, a friend offered to loan me his operating system disk so I could get started. It didn't work. However, the system reported the disk had checksum errors on every track. I tried it several times without success, then took the disk to the local Heath store, put it in their system, and it ran perfectly.

After several hours of retracing my steps (all the tests in the operation manual checked out, every time), I took the system to my friend's house to try interchanging components to isolate the problem. My drive did not work with his controller board. We examined the programming plug in the drive, and bent the cut jumper strips away from each other, just in case they might have been touching. (The programming plug, with the proper jumpers cut, identifies the drive number to the system.) There is also a small plug on the drive circuit board where the read/write heads connect to the board. I'd encountered a lot of problems with such plugs in the past, perhaps from metal oxidation, loose connections or something, so I wiggled the plug on its pins just before we turned the system on again. It was an incidental act, but it corrected the problem. The system ran perfectly.



H-17 Disk Drives, H-8 Computer, Selectric (modified), Hazeltine 1500 terminal.

The startup procedures are complicated, but if one is careful to follow the instructions, they are clearly spelled out. In the Diagnostic Program (TEST 17), "General Checkout" takes about 45 minutes to run, although it requires no operator attention. My system had no problems, so I don't know what more would have been required had there been problems.

HDOS — The Heath Disk Operating System

Having fought with cantankerous cassettes for over a year in my Heath as well as another computer I have, I am delighted with the speed and reliability of the floppy disk way of computing. The first week was pretty frustrating, trying to remember the

subtle but important differences between disk system commands and cassette system commands. But I haven't lost any disks yet, and my productivity is picking up.

Under HDOS, one can address any file in either disk drive, copy, console display, or print it through the single output "device," the Alternate Terminal. Files can be accessed and manipulated by BASIC, EDIT, DBUG, or ASM (assembler), regardless of the originating program. For example, a text file generated in EDIT can be reformatted by a BASIC program, to change line length or justify right margins, or whatever. BASIC programs can be loaded into EDIT for alteration. I should think it possible to write a BASIC program to create an assembly-language program from various parameters, if one is so inclined. HDOS will permit it.

One can even create a text file directly through HDOS, without the Editor (EDIT), by "copying" file "TT:" (the console keyboard) into a named disk file. In the same way, text or a BASIC program can be printed through HDOS without having to load EDIT or BASIC.

They haven't forgotten that most H8 users have cassette files and programs, and so have included special HDOS programs for copying from cassette onto disk files. Text and most BASIC programs may be transferred, although some cassette BASIC statements must be changed to allow them to run under HDOS. Machine language programs cannot be transferred, but assembly language source code can be copied and reassembled, with some limitations in I/O and memory use. So far, there's no way to copy disk files to cassettes.

The peculiar memory allocation that Heath started with — the first 8K is used by the front panel routines, HDOS ROM and RAM, and some, as yet unassigned but unavailable to the user — discouraged some adaptations of other software for use in the H8/H17. Under cassette operation, all machine language programs were expected to originate at 040:100 (split octal); under HDOS they start at 042:200. About 2K at the top of memory is used by HDOS, as well. In my machine, with 16K of RAM, I have about 3K available on top of BASIC, and over 8K on top of the Editor. The editor will run with open files for input and output, so text length is limited only by disk capacity. If one works in BASIC very much, however, 16K is apt to put pretty tight restrictions on what can be done.

The principal HDOS functions are shown in Table 1. The system disk also includes BASIC, EDIT, ASM, DBUG and a couple utility programs for loading cassette files.

HDOS FUNCTIONS

BOOT	Reads the system disk, enters Command Mode, asks for date (automatically dates all new files).
CHECK	Reads the disk, checks for damage to files.
INIT17	Initializes new disks with name and number.
TEST17	System diagnostic, checks drive speed, read/write functions, media (disk) condition.
SYSGEN	Copies HDOS system onto another disk.
ONECOPY	Copies any disk file to another disk using only one drive.
CAT	Displays a catalog listing of files on a disk.
TYPE (or LIST)	Types out the contents of any file on the console.
RUN	Executes any machine-language program.
COPY	Copies any file, to another file, to the console or to Alternate Terminal (printer).
RENAME	Renames any file.
DELETE	Deletes any file.
SET	Sets any of a number of system options, such as drive speed, maximum line length, etc.
FLAG	Affects write protection for a file, or suppression of its listing in CAT, or locks it from any change or deletion.
PATCH	(Unexplained in manual).
STAT	Reports number of hard and soft errors encountered since BOOT.
DEMO	Actually four demonstration programs, in assembly language and BASIC.
ERRORMSG	Any of 69 different error messages.
HELP	Displays list of options in command mode.

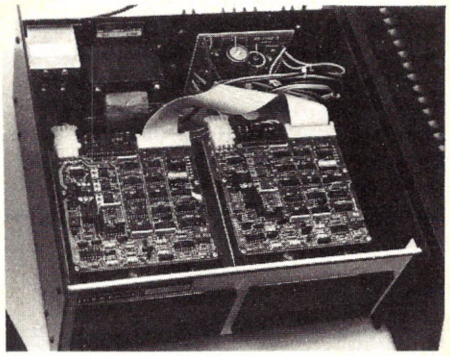
Basic

Like Heath's cassette version of extended BASIC, this permits multidimensioned matrices of variables, including string variables. (Up to eight subscripts are permitted per variable). Strings may be up to 255 characters long. I/O operations include PEEK and POKE, to read or write values into memory, PIN and OUT to read and write to a port, and five channels to use for reading and writing to disk files. These channels can also be assigned to the console and to the Alternate Terminal, for hard copy output. A program can be SAVED, and an updated version can REPLACE an existing one in a disk file. FREEZE will store a program, its variables, and BASIC itself, into a special file in case it's three a.m. and you're too tired to finish running it.

INPUT and LINE INPUT work the same as with cassette BASIC, except that they can be used to read from a disk file, as well as from the keyboard. Strings can be read without quotation marks through LINE INPUT, allowing unlimited access to text files as "data."

The function CIN(n) is new to Heath BASICs. It examines the next character available at the input file (n = channel number). With some experience and a few hints, it can be very useful. For example, disk files are stored in sectors; unused space in a sector is filled with zero values. The end of a text file then can be identified by the first zero encountered, flagging the end of a read operation. Another good use for it is to read input one character at a time, as when formatting text, to look for spaces, carriage return characters etc.

BASIC programs can be CHAINed together, with variables and open files saved, permitting very long programs to be broken up into parts that will fit the available memory, or to pick up frequently used routines instead of repeating them in every program. The CHAIN command, which can be included as a program statement, calls up another BASIC program and tells it to execute using existing variables and files.



Inside the H-17 drive, showing the two Wangco drives and power supply (at rear). Open design could accommodate other drives. (HDOS can support only two drives).

The LOCK command prevents data and variables from being cleared, and also prevents program text from being altered or deleted. This could be valuable in situations requiring protection of data not existing in a disk file, such as business accounting data during processing.

The most noticable deficiencies in this BASIC that I've encountered are: the lack of the USR function, that calls a machine-language program; no automatic renumbering of statements; and that to edit a statement you must retype the entire line (although you can edit strings in the Editor).

Disk System, con't...

Editor

The Heath line-oriented editor is adequate for writing assembly language programs, but cumbersome for text. It's essentially the same as the cassette version TED-8, but with disk file access it's handier to use. Actual editing of copy takes a lot longer than a writer wants to spend.

A new text editor is supposed to be released by Heath, however; perhaps it will be designed more for text than programming.

Assembler and Debugger

The most notable feature of ASM is its ability to pick up program segments from disk files, as it assembles. Frequently used routines or symbol definitions, then, can be written once, and included in any subsequent program. CALLs to HDOS routines may be entered symbolically, and HDOS will supply the necessary addresses, as well as the routines themselves. HDOS also maintains a "sector cursor," which can be moved to any 256-byte sector in a file for reading or writing, providing random access to file contents.

The console debugger, DBUG, allows a machine language program to be loaded, run, stepped, examined or changed from the console. Registers and memory locations can also be examined and changed at any point. Values can be displayed in decimal, octal or hexadecimal notation.

Manuals

Heath is justifiably famous for their assembly manuals. The disk system assembly manual is as good as any of those I've used before. They include several pages of changes which had to be inserted before I could begin construction, but that is expected with any new product. They really lead you by the hand through assembly, and leave very little to chance.

The operation manual, as I mentioned earlier, goes only far enough to check out the system; it really says very little about operation. The procedure for testing was garbled, and it took several hours to figure out what they were trying to accomplish. This is unusual for Heath, and I hope they have corrected it by now — I complained about it at the time.

Once the system is running, the Software Reference Manual is the operating manual. The other two can be stored away, along with the little booklet that comes with the Wangco drive.

I found the manual hard to use as a "reference manual;" indices in the various chapters were inadequate (the HDOS section is 120 pages long, but the index contains only 71 items!). They've printed edge marks like tabs in the BASIC section, but I still have trouble finding particular commands. There is a summary list of commands in an appendix, but it runs 12 pages long. I finally typed my own "command index," and put it in the front of the BASIC section, where it's instantly available behind the tab divider.

There's another manual available for HDOS, a System Programmer's Guide, which includes information needed by anyone intending to write assembly-language programs that will interface with HDOS. On the first page, it warns that it is "not an official Heath manual," and is not written in a tutorial manner. It was, however, obviously written by someone who knows HDOS forward and backward and, while the jargon gets a bit thick in places, so does the colorful description: "Do not single step through a HDOS SCALL, you may screw up the disk." To the point and authoritative. That manual is available through the Heath User's Group, for $5.00.

Alternative Software

Heath has also announced the availability of two commercial software packages running under HDOS: MICROSOFT (tm) BASIC and MICROSOFT (tm) FORTRAN. Among the additional features of the MICROSOFT BASIC not in the Heath BASIC are IF-THEN-ELSE statements, automatic line renumbering, double-precision math, and a PRINT USING statement for formatting output. This BASIC costs $100 extra, and requires 32K of memory. FORTRAN is one of the few available for 8-bit computers, costing $150 and requiring 40K of memory.

Building on the MICROSOFT BASIC, they are offering a set of business application programs, including General Ledger, Accounts Payable, Accounts Receivable, Payroll and Inventory packages. The cost of these programs was unavailable at this writing, but they were scheduled to be published in the June catalog.

A kind of "do-it-yourself" text editor has been offered for sale by the Heath User's Group. Based on a public-domain editor donated to the CP/M User's Group, it is character-oriented, rather than line-oriented, and so is much faster to use for paragraph material than is EDIT. Although it lacks some functions (such as output to a printer), they furnish all the source code with it, so one can write output routines, alter existing ones, and generally patch it up to suit the need. Even the instruction manual is included on the disk rather than being printed, so the whole thing is a kit-builder's kind of software. Members can get it for $15. Other software, contributed by the members themselves, is also available for the cost of publication.

Conclusion

Since I have two computers (the other one is a Xitan, with S-100 bus), I did a lot of thinking about which disk system to buy. There's security in having a standard configuration, and several good systems are available for the S-100. But I've found that there's also security in a single source. Heath has been very good about supporting their products, as long as they're all Heath. (I had problems last year trying to interface a printer to my H8, but it's working now.) I chose to stay with Heath because of that support. If my S-100 disk system wouldn't run, would the problem be in the drive, the controller, the CPU, the motherboard or the software? And who's going to help me?

I hope Heath (or somebody) makes available some more software, particularly for word processing, but the H8/H17 system runs well, forgives a lot of my fumbles, and hasn't wiped out a disk yet. I think I'll keep her. □

HEATHKIT PRICES (Mail Order)

H8 Computer, with 16K memory and a terminal/cassette interface board	$899
H9 CRT Terminal	$479
H17 Disk System, with one drive and HDOS software	$675
Second Drive (assembled-just plug in)	$295
Four-port serial or three-port parallel I/O board	$150
H14 Line Printer (requires Four-Port Serial Board)	$595
Additional 16K Memory (wired)	$395

(prices from spring 1979 catalog)

...so, one can start with an outlay of $2053 plus postage to get a system without a printer; $2798 with printer.

*Lifeboat Associates (164 W. 83rd St., NY, NY 10024) is offering a version of CP/M that will run in the Heath Disk System. CP/M will open a very large amount of software to Heath users, from sophisticated word processors, to highly specialized business programs.

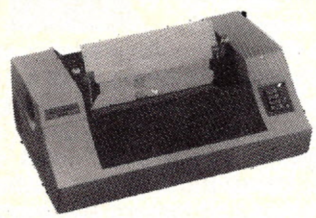
The secret of success in conversation is to be able to disagree without being disagreeable.

USED

CENTRONICS PRINTERS

IMMEDIATE DELIVERY ★ 30-DAY WARRANTY
UNITS ARE FACTORY-REFURBISHED ★ SOME NEW

MODEL 102A

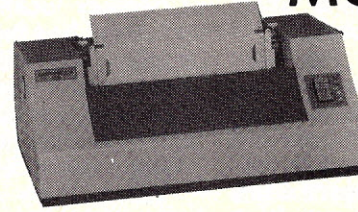


$1,190

- 330 CPS, 132 Columns
- 5×7 Dual Printhead
- Bidirectional Printing
- Multipart Forms Capability

+ 120 lbs. shipping

MODEL 103

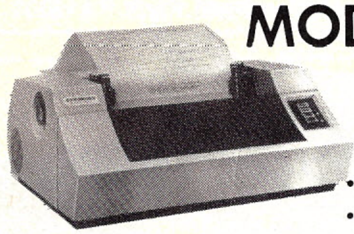


$1,695

- 165 CPS, 132 Columns
- 5×7 Printhead
- LSI Circuitry
- Bidirectional Printing

+ 120 lbs. shipping

MODEL 101A

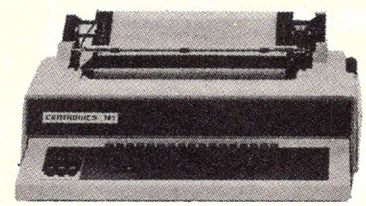


$925

- 165 CPS, 132 Columns
- 5×7 Printhead
- Manual Line Feed
- Remote Select/Deselect

+ 120 lbs. shipping

MODEL 761 KSR



$795

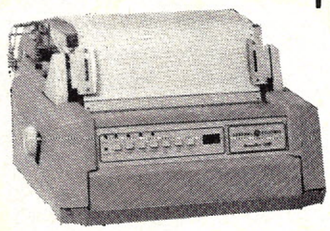
- 60 CPS, 132 Columns
- 7×7 Printhead
- ANSI Keyboard
- Bidirectional Printing

+ 75 lbs. shipping

LOTS MORE CENTRONICS TO CHOOSE FROM . . . OVER 29 DIFFERENT MODELS IN STOCK

GENERAL ELECTRIC

High Reliability TERMINET SR300



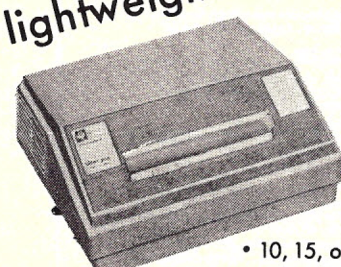
READ ONLY **$700**

- 10, 15, or 30 CPS
- High quality Belt Printer
- RS232C Standard
- 80 Char. line
- Adjustable Tractor feed

+ 50 lbs. shipping

lightweight!

T.I. 755 THERMAL PRINTER



$595

- 10, 15, or 30 CPS
- USASCII Code
- RS232C Standard
- 80 Char. line
- 8½ inch friction feed platen

+ 25 lbs. shipping

HOW TO ORDER:

1. Send Check, Money Order, or Credit Card # (M.C. & Visa)
2. Be sure to include shipping charges. Contact local agent for estimates on truck, padded van, or air freight costs.
3. Enclose Name, Address, and Business and Home Phones.
4. Mass. Residents add 5% Sales Tax.

A AMERICAN USED COMPUTER

P.O. BOX 68C, KENMORE STATION, BOSTON, MASS. 02215

FOR MORE INFORMATION CALL 617-261-1100

Instructo Paper Computer

Randy Heuer

The Instructo Paper Computer (IPC) retails for $5.95 plus postage and proclaims to be a real programmable computer. Well, I'm not sure that those people with an IBM 370 are quite ready to trade their machine for an IPC, but if you are interested in learning something about the internal operations of a computer, you may want to give IPC a try.

The IPC is made of cardstock just as the name implies. The basic unit measures 16" x 19". Several paper slides for the jump switches, compare unit, program step indicator, index counter, registers, inputs and outputs are provided. The unit requires a few minutes to punch out and assemble. Power for the unit is provided by the operator's fingers and a writing implement, i.e., a pencil. A 32-page Operator's Manual is also provided.

The IPC is primarily intended as a tool for teaching the basics of computer operations. IPC uses a pseudo-assembly language. Rather than confuse the operator with octal or hexadecimal codes, multiword instructions and the like, each instruction and memory location takes up one location in the computer's base ten program storage. Computer instructions are in the form of mnemonic assembler codes. For example, if you encountered a LDRA, 13 (load register A with the contents of memory location 13) instruction, you would find the number or word in location 13, follow the data bus arrows to the register A paper slide, write that number or word on the register A slide and then advance the program step indicator by one.

About thirty different computer instructions are covered in the manual which also contains sixteen sample programs. Of course, the operator can write and run his own programs.

The IPC won't quite prepare you to begin to write assembler code for real electronic computers, but offers an excellent first look into the "mysterious" insides of these machines. Using the IPC will probably make things a bit less mysterious and allow you to have a little fun while learning. For further information, write to: INSTRUCTO/McGraw-Hill, Cedar Hollow Road, Paoli, PA 19301. □

CBASIC - A Review

James McClure

If there is one aspect of the microcomputer industry that has not kept pace with the others, it must be software. However, it's wrong to say that high level software isn't available — you just have to look a little harder. That's how I found CBASIC, a BASIC language compiler/interpreter developed by Software Systems Inc. for 8080 and Z-80 based microcomputers.

Soon after I purchased my computer, I realized that a fully versatile system required a floppy disk, so I bought one, along with the CP/M disk operating system from Digital Research. Both have proven to be excellent. However, as with most operating systems, CP/M does not come with a high level language, so I went looking again. While turning the pages of a computer magazine, I spotted a small ad for a new BASIC compiler which would run under CP/M. It sold for only $99—less than one third the cost of its closest competitor. I sent a check and received a comprehensive manual along with a diskette containing the language in about ten days. This was to be the beginning of a long and happy relationship.

Overview

In the course of assembling my system, I went through six BASICs (including one I wrote myself) and decided that CBASIC had several main advantages over the others.

It's ANSI standard BASIC. While BASIC may not be the greatest language, it is the most popular, which means you will be able to make use of a large reservoir of applications programs. Since it's ANSI standard, you'll be able to swap programs written for different computers without having to do a lot of rewriting. CBASIC is also completely disk oriented, communicating through CP/M with the files residing on the diskette. Any file may be accessed through CBASIC commands, regardless of its type or location. Full file facilities are provided, including random and sequential access methods; several other useful commands are also available for general housekeeping tasks, such as deleting and renaming files, etc. Furthermore, since disk is the major storage medium, programs, data and text are available quickly and reliably.

James McClure, 1019 Van Kirk St., Philadelphia, PA 19149.

Operation

Unlike conventional BASICs, CBASIC is implemented as a hybrid language—it's half compiler and half interpreter. This means that the steps involved in writing a CBASIC program are quite different from those used with a simple interpreter; to be precise, they are more complicated.

With a straight interpreter, the user simply enters the desired program while in the BASIC's command mode. Line numbers precede each program line and tell the computer in what order the lines are to be executed. When the entire program has been entered, it can be immediately executed by typing "RUN" or some similar command. If mistakes are found, they can be corrected through a limited editing mode, and the program can be resumed with no further delay. These facts make BASIC interpreters excellent for interactive program development.

On the bad side, interpreters must evaluate program statements each time they are executed, making them rather slow. In order to combat the speed problem, most BASICs place some limitations on the method in which the program is written. For example, in order to keep the time required for the computer to recognize a variable name down to a minimum, only the first two letters are considered. This also must be done to save space, another problem with interpreters, since the entire source listing (including blanks, remarks and other things which are useless to the computer) must reside in memory. Still other hassles result from the requirement that every line must be assigned a number, even if that number is never referenced by another program statement. Furthermore, once numbers are assigned, it is impossible to rearrange portions of a program; it must remain in the same order.

These problems, inherent to almost every interpretive BASIC, do not exist in CBASIC. Variable and function names can be up to 31 characters long and may contain keywords with no confusion on the part of the compiler. Lines may be continued indefinitely, and each line is not required to have an associated number. Remarks take no space in the compiled program code, so they may be used often, greatly aiding program documentation. This is not so important in programs used only on a private system, but who knows when your next programming masterpiece may be sold for general distribution? Also, many users' groups sponsor software exchanges, and documentation is an important factor.

How are all these benefits possible? As the saying goes, "you don't get something for nothing," and programming is no exception. These advantages of CBASIC have been implemented at the cost of interactiveness. Unlike a traditional interpreter, CBASIC requires that a source file containing the program statements be created using the system editor. This can be anything from the standard editor supplied with CP/M, called ED.COM, to a fancy word processing editor. Once the file is created, it must be compiled into an executable intermediate file by the program CBASIC.COM. Once this step is completed, the intermediate file is loaded and executed by the CRUN.COM program. Thus, running a program with CBASIC is accomplished in three steps. The source file is edited, then compiled, and finally executed.

Most of the advantages of CBASIC are made possible by the compilation process. It is during this phase that the variable and function names are evaluated, and addresses assigned for each variable. This saves time when the program is executed, since the run-time monitor does not need to re-evaluate each identifier as it is encountered. Also, during compilation, blanks and remarks are filtered out and keywords are compressed. String and numeric constants are evaluated, thereby relieving the interpreter of costly decimal-to-binary conversions, and saving more time. Finally,

CBASIC, con't...

the labels are resolved and the actual program pseudo-code is generated. The psuedo-code and the other program elements are stored in an intermediate file, considerably smaller than the original source, which will be executed by "CRUN.COM."

The pseudo-code itself, abbreviated P-code, is an interesting topic. P-code is a cross between machine code (the binary instructions recognized by a given microprocessor) and higher level statements, such as those encountered in BASIC. Generally, a BASIC, PASCAL or COBOL compiler defines a minimal set of instructions which will be needed to implement the language. These instructions may include move commands, floating-point arithmetic operations and others. Once these instructions are defined, a hypothetical "machine" capable of executing them is created. However, rather than building an actual microcomputer capable of executing the desired instructions, a program is written for one of the already available machines (8080, Z-80, 6502, etc.) which interprets the P-code, and performs the necessary operations. Hence comes the notion that CBASIC is part compiler and part interpreter.

There are many advantages to P-code interpreters, sometimes called P-machines or virtual processors, which are beyond the scope of this article to discuss. The reader should be aware, nonetheless, that CBASIC employs this general type of system.

As mentioned, the compilation process is not present in the operation of a normal BASIC interpreter; as a result, programs written in CBASIC generally take longer to debug, since the source must be recompiled after every modification. Once the program is finished, however, it can be run any number of times without the need for recompilation.

Features

CBASIC is one of the most feature-packed languages available to computer owners. In terms of arithmetic capabilities, it possesses a full complement of math functions (see Figure 1a) as well as the ability to work with both real and integer variables; real quantities are stored with 14 digits of precision and exponents of ten varying from 63 to -64. Integers range from -32768 to 32767. Both hexadecimal and binary constants may be used in addition to the usual decimal numbers.

In addition to numbers, CBASIC can also process strings of characters. String variables may contain from 0 to 255 characters, and may be manipulated through a variety of functions (see Figure 1b).

Operators: + , -, /, *, v (exponentiation), AND, OR, NOT, XOR (exclusive OR)

Trancendental functions: SIN, COS, TAN, ATN, LOG (natural), EXP

Conversion functions: ABS, INT, FLOAT

Machine code functions: INP, PEEK

Others: FRE, RND, SGN, SQR (square root)

a. Math Functions

Operators: + (concantation)

Substring functions: LEFT$, RIGHT$, MID$, MATCH$ (searches for substring in specified string variable)

Conversion functions: ASC, CHR$, UCASE$ (converts argument to upper case), STR$, VAL

Others: COMMAND$ (returns CP/M command line), SADD% (returns the address of the specified string variable)

b. String Functions

Figure 1. CBASIC Functions.

Along with the standard ANSI BASIC instruction set, CBASIC also provides a number of special commands which make programming easier. The first of these special commands deals with structured programming which involves the construction of programs out of blocks. Each block performs a given task within the program. The blocks are usually implemented as subroutines, known in some other languages as procedures or paragraphs. The pros and cons of block oriented, or structured, programming have been debated for many years; it is generally agreed, however, that this technique increases the readability of programs, as well as making them easier to modify.

As an example of the structured technique, suppose that you are asked to write a program which is to read names from a given disk file, sort them, and then write them back to the file. Using the structured method, the program would be logically divided into three distinct parts: a block to read the disk file into memory, a block to do the actual sort, and a block to write the sorted information back to the disk. From this initial conception, the three modules could be designed, programmed and debugged separately, and then combined to form the final product.

Some languages, such as PASCAL and PL/M, were designed with structured programming in mind. Unfortunately, this is not true of BASIC. Thus, many of the commands which make structured programming possible in other languages are not present in the ANSI standard of BASIC. In order to get around this, and still offer programmers a choice, CBASIC has been expanded to include some structured commands.

The first of these is the WHILE command. It is used in situations where a group of instructions must be executed WHILE a specified condition remains true. For example, suppose that the elements of array N are to be printed until a key on the console is pressed. Figure 2a lists a standard BASIC program written to perform this function. In it, line 110 tests the console to see if a character has been sent. If so, the computer branches to line 150 and STOPs; otherwise the next element of N is printed, and the computer is sent back to 110 by the GOTO at line 140. Listed in Figure 2b is the same program written in CBASIC, using the WHILE command. Notice in this program that the WHILE and WEND commands denote the loop boundaries. In this form, the routine can practically be read as an english sentence, making it self-documenting in the sense that many REMarks are not necessary. When used frequently, the WHILE command can greatly improve the readability of programs which might otherwise be made incomprehensible by a tangled mass of GOTOs.

```
100 E = 1 : REM Set pointer to first element.
110 IF CONSTAT% THEN 150 : REM Test
    console.
120 PRINT N(E) : REM Print next element.
130 E = E + 1
140 GOTO 110
150 STOP : REM Terminate if a key has been
    pressed.
```

a. Standard BASIC Program

```
ELEM = 1
WHILE NOT CONSTAT%
   PRINT N(ELEM)
   ELEM = ELEM + 1
WEND
STOP
```

b. CBASIC Program with WHILE Statement

(CONSTAT% is a function which returns the value TRUE when a key on the console has been pressed, and the value FALSE otherwise.)

Figure 2. Loops in CBASIC

Another handy feature useful for structured programming is the multiple line, user defined function. A function is the same as a subroutine except that it returns a value. Traditionally, BASIC functions which are defined by the user have been restricted to one line. For example, the function of Figure 3a is used to calculate the logarithm of a number N in base A.

This format is acceptable when the function is short and simple. In some cases, however, it may be more desirable to implement an entire block of a structured program as one function. CBASIC has an extension to allow user defined functions to have

CBASIC, con't...

more than one line which means complex operations can be programmed as functions. As an example, let us go back to the previous assignment; we need a program to sort the items in a disk file. The first block of the program will read the elements of the file into array STRINGs. Figure 3b shows this block written as a multiple line function.

```
DEF FNLOG(X,A) = LOG(X)/LOG(A)
    a. Single Line Function
DEF FN.READ.FILE% (NAME$, MSIZE)
    FN.READ.FILE% = FALSE%
    RECORD% = 0
    IF END#1 THEN 110
    OPEN NAME$ AS 1
    FOR RECORD% = 1 TO MSIZE
     READ #1: STRINGS$(RECORD%)
    NEXT RECORD%
110 IF RECORD% = 0 THEN RETURN
    FN.READ.FILE% = TRUE%
    RETURN
FEND
    b. Multiple Line Function
```

Figure 3. CBASIC User Defined Functions.

Notice that the file name and the maximum number of entries which can be sorted are passed to the function as parameters. An unlimited number of other parameters could have been passed, all of which are substituted for the dummy variables shown in the first line of the function's definition. As you can see, the parameters may be of several types (i.e., strings, numbers, etc.).

Once the main programs has passed the necessary information to the function, the statements of the function are executed, which, in this case, causes the contents of the file whose name is stored in NAME$ to be read into the string array STRINGS$. When the function completes, it returns a value back to the main program. In this example, the value returned is either TRUE or FALSE; if TRUE is returned, the read operation has succeded; if FALSE is returned, there has been an error.

One of the more important features of multiple line functions in CBASIC is their generality. For instance, a set of desired operations can be implemented as functions, which are then stored in a library file on the disk. Assuming no line numbers are referenced within the functions, they can then be included anywhere in any source program, and utilized as if they were a built-in part of the BASIC. CBASIC even has a special compiler directive which facilitates this application; the %INCLUDE command. This command, followed by a filename, causes the compiler to take the contents of the specified file and INCLUDE it in the text of the source program being compiled. Therefore, once a set of custom functions has been developed, other programs can make use of them freely. Furthermore, it is easier to debug a number of individual routines than to debug a single, massive program.

One structured programming feature noticeably missing from CBASIC is the nested IF. Most BASICs permit IF statements to contain other IF statements, useful in processing multiple conditions. For example, the line shown in Figure 4a, while acceptable in other BASICs, is illegal in CBASIC. However, often the statement can be rewritten in another form, such as that of Figure 4b. I spoke to Mr. Gordon Eubanks, author of BASIC-E, CBASIC, and CBASIC-2 (a prolific programmer!) regarding this point, and he told me that the next major edition would be changed to permit nested IFs.

```
IF LIGHT = GREEN THEN
  IF TIMER = 50 THEN GOSUB 10000
    a. Nested IF Statement

IF LIGHT = GREEN AND TIMER = 50
  THEN GOSUB 10000
    b. CBASIC Compatible Version of (a)
```

Figure 4. Multiple Condition Statements.

Moving away from the structured programming arena, there are several other special commands which may be of interest to machine language programmers. PEEK, POKE, INP and OUT statements allow direct access to memory locations and I/O ports, and a CALL statement has been included, so that BASIC may branch to machine code routines anywhere within the computer's address space. In my opinion, this CALL command is by far the easiest and most uncomplicated solution to the machine language linkage problem that I have seen in any BASIC.

If desired, the SAVEMEM command may be invoked to reserve room at the top of the CRUN interpreter space. This command also provides for the loading of a disk file containing the assembled machine code programs into the reserved area. This is particularly handy for people interested in analog and control applications, since the input/output drivers for analog and digital interface boards are generally easier to write in assembly code. CBASIC allows such routines to be easily combined with BASIC programs.

For those interested in writing data base programs, CBASIC has a good assortment of disk file commands. Both random and sequential access are supported. Disk statements include OPEN, CREATE, DELETE, RENAME, READ, PRINT, CLOSE, and SIZE.

All files are read and written as a stream of ASCII characters, which can be a drawback, since numbers must be converted from internal representation, to ASCII, and back to internal form. Perhaps a good addition to CBASIC would be a means to write variables directly to the disk in their internal format, thereby saving time and space.

One of the more notable features of CBASIC's random file scheme is the fact that record length is not fixed to the physical sector length (128 bytes). Through the use of the RECL command, record lengths can be defined without regard to physical disk boundaries, freeing the programmer of many headaches, and conserving space on the diskettes. Furthermore, since CP/M allocates disk space dynamically, the size of a file does not need to be determined at the time it is created. Thus, a mailing list file is free to grow to fill all available space without the need to rearrange other files.

Benchmarks

When a new language is released, one of the questions most often asked by computer enthusiasts is "How fast is it?" Now, in all fairness, speed is not the most important aspect of any system. After all, the object of the computer is to solve problems. In the first part of this article, I discussed the commands which have been added to CBASIC to make programming (and problem solving) easier. To keep things fair, I ran the seven benchmark programs designed by Tom Rugg and Phil Feldman (See "BASIC Timing Comparisons," KILOBAUD, June 1977, pp. 66-70) under CBASIC (and the more recent CBAS2 release) in order to determine how fast the language is. All calculations were done in 14 digits, which constitutes double precision in most BASIC languages. CBASIC finished considerably behind most of the other languages listed in "BASIC Timing Comparisons," even taking the double precision into account. The results are shown in Table 1.

The benchmarks were then run under CBASIC version 2, and, surprisingly, proved this later version to be even slower than its predecessor, and by a good 63% (on the average). Even when the program was run using integer variables (BENCHM2) in all loop constructs (as would normally be the case), the results were still far from thrilling. Here is an area which definitely needs improvement, since it appears to me that the speed could be improved at least somewhat,

CBASIC Timing Results

Program	Benchmark Number 1	2	3	4	5	6	7
BENCHM1 (CBASIC)	6.1	6.9	52.3	60.5	61.0	88.2	120.7
BENCHM1 (CBAS2)	8.1	16.0	60.9	96.3	96.8	163.0	192.2
Change	+33%	+132%	+16%	+59%	+59%	+85%	+59%
BENCHM2 (CBAS2)	2.8	3.4	15.0	61.6	62.0	76.9	94.5

Notes: All times are in seconds.
All non-integer calculation was done in double precision (of necessity).
BENCHM2 is a version of BENCHM1 which uses integers in all loop constructs.
All programs were run on an Altair 8800A with 2MHz clock and static memory with no wait states.

Table 1. Benchmark Speed Results.

without curtailing any available features. Hopefully, future releases and versions will work towards this goal, since the other aspects of CBASIC make it so worthwhile.

The Report Card

Overall, I would have to rate CBASIC highly. It's $99 price tag makes it one of the more inexpensive high level programming languages available, and the fact that it is compatible with CP/M means that it can be used on a large variety of systems. My opinions on CBASIC come from extensive use, rather than from the reference manual which, incidentally, is quite comprehensive, though not directed towards the beginner. C-BASIC has been a valuable asset to my system and I feel that it is a worthwhile investment for any computer owner interested in developing his own applications software with a minimum of fuss and bother.

CBAS2 may be obtained by sending $99 to: Software Systems Inc., Box 145, Sierra Madre, CA 91024. Owners of the first release of CBAS2 may obtain the updated release (CRUN 2.04, and CBAS 2.02) by mailing $5 to the company. Discounts are also available to owners of CBASIC version 1 who wish to purchase the newer version.

In conclusion, I have found CBASIC to be a most useful tool, and I urge other computer owners to look into it as a means of expanding the usefulness and versatility of their systems. □

The question "Who ought to be boss?" is like asking "Who ought to be the tenor in the quartet?" Obviously, the man who can sing tenor.

Henry Ford
Forbes Magazine

A Dozen Apples for the Classroom

Joyce Hakansson
Leslie Roach

The Science Shuttle is a unique and innovative project designed to bring computers into the school classroom. Each morning a van loaded with 12 Apple II microcomputers leaves the Lawrence Hall of Science for a school somewhere in the greater San Francisco Bay Area. Once there, two instructors unload the micros and in a half-hour set up a temporary computer laboratory within the school. During the day up to 120 students will experience the joy and excitement of learning and creating programs using interactive computers. For some students this might be a one-time introductory workshop, while for others it will be one in a series of programming and problem-solving classes.

One advantage of the Science Shuttle is that it allows us to demonstrate in local classrooms the teaching approach used at the Lawrence Hall of Science. The Lawrence Hall of Science is a science museum and a science education research center located on the University of California, Berkeley, campus. The teaching approach at the Hall is to provide people with "hands-on," participatory experiences that will promote discovery learning. For computer education, this means that the students have a chance to create, write, and debug their own computer programs while working at a computer terminal.

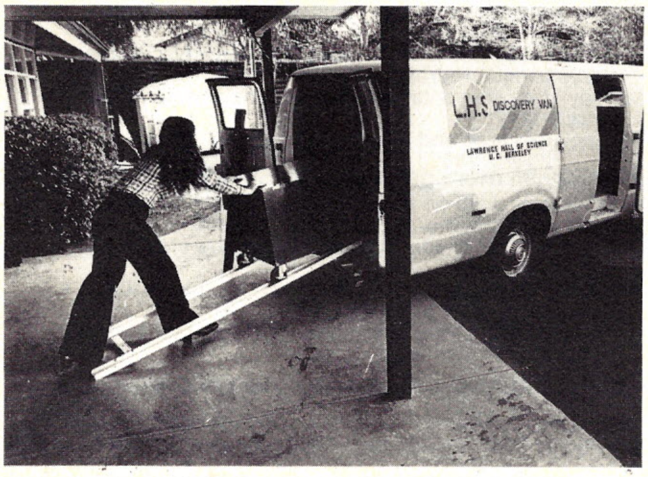
For the past eight years LHS has pioneered hands-on computer education through activities based on our 80 terminal time sharing system. Terminals are used as exhibits in the museum, in classes at LHS, and casually by individuals coming to the Hall. We log annually about 40,000 paid enrollments in our computer activities.

A few dozen Bay Area schools are remote users of the time sharing system, but almost none can afford to have more than one or two terminals, and cannot use our hands-on approach with a full class of students. The Science Shuttle now makes it possible for the first time to take this approach with us out to the schools. There the instructors, who are University undergraduates from various academic disciplines, encourage students to view the computer as an intellectual tool. Students are taught to develop a problem solving strategy from which they devise a logical procedure for the computer to follow. They then translate their description into a programming language, BASIC, which is understood by the machine. Debugging can be tedious, but the prospect of sharing a well-written program with others is a tempting challenge.

The Science Shuttle has allowed the Lawrence Hall's Computer Group to realize two of its long term goals: to make computers accessible to a larger number of students and to bring computers into the "average" classroom. Due to cutbacks in State funding for educational programs, many schools found that they did not have money in their budgets to transport students to us. Even when funds were available the number of classes we could teach at the Hall was severely limited by the

Joyce Hakansson, Coordinator, Computer Education; Leslie Roach, Computer Group; Lawrence Hall of Science, University of California, Berkeley, CA 94720, (415) 642-3167.

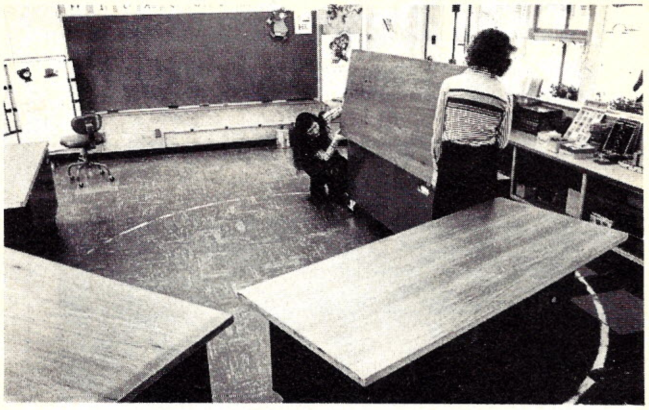
Leslie Roach and Margie Gardner show off one of the Lawrence Hall of Science "Apple Carts" and its contents as they pack up the Science Shuttle for its daily trip to a San Francisco Bay Area school. The "commuting classroom" has been booked solid since it began operation in January.



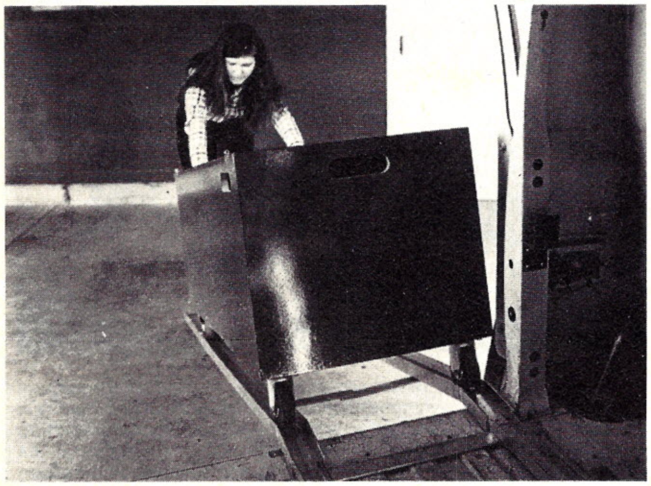
The van accommodates four carts. Each one contains three computer systems consisting of an Apple II computer, a disk drive and a color TV that serves as a monitor. Unloading the van is a one-person job. Each cart weighs about 250 lbs. when fully loaded, but the ramp and large wheels make it relatively easy to unload and wheel each one into a ground-level classroom.

size of our facility. Once portability was made possible by the microcomputer, it was only logical for us to bring our classes to schools unable to come to us. Partial support from Apple Computer Inc. has made it possible to establish the Science Shuttle.

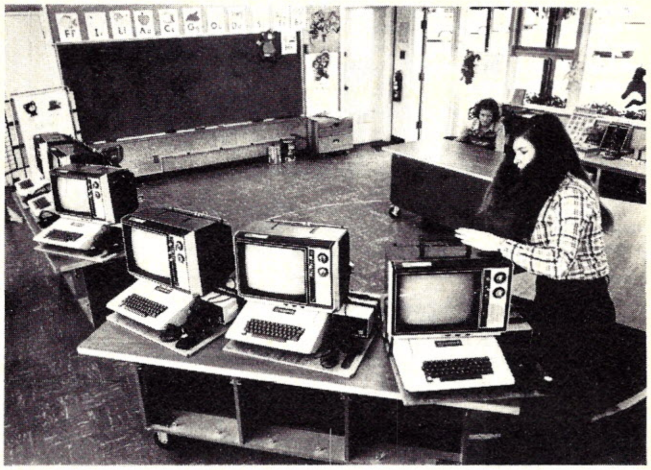
Many educators are aware of the importance of computing, but few of them know how to bring it into the schools. The Science Shuttle is a total computing package (equipment, instructor and curriculum) at the relatively low cost of about $200 per two-hour visit. It has been a colossal success, with nearly every date booked from January 1 to the end of the school year — 6000 miles of travel and 5000 contact hours. The students enjoy the experience, the teachers become excited about it as a new teaching method, and administrators have a chance to see computers in the classroom before making a large capital investment. Often this combination has led schools to initiate their own computer education programs. □



A positive aspect of the program is its portability. We can teach almost anywhere. This happens to be a classroom, but we have also used libraries, multi-purpose rooms, and even outdoor patios. The carts form the base for a work station. Hollow-core doors are hinged onto the top allowing desk space for the students.

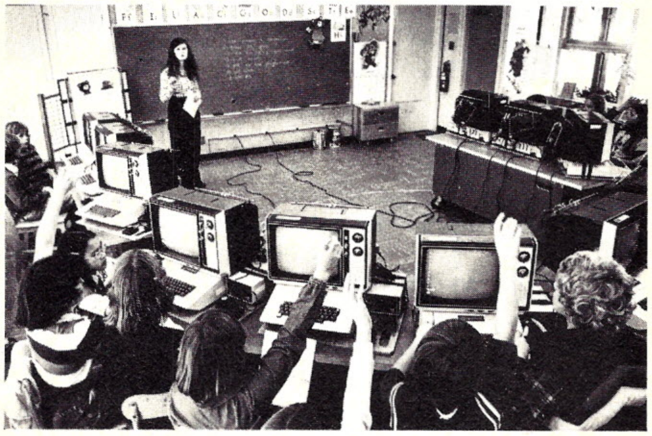


Here goes the last of the four carts. Going out with the Science Shuttle tests an instructor's versatility. Leslie drives the van, unloads the carts and teaches the classes. If any of the equipment malfunctions, she has learned to diagnose and, in many cases, fix the problem right there. Despite all of the travelling, the equipment has held up very well.



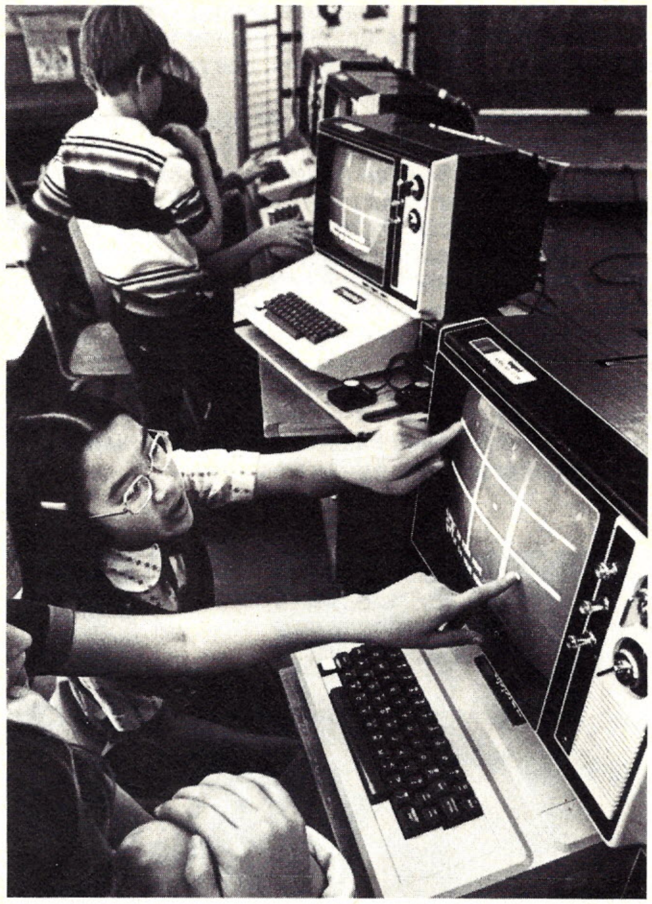
Each cart with its table-top accommodates three microcomputers. The board containing an Apple and its disk drive is placed on the table. The 13" TV monitor fits easily on top. Not much more has to be done, and it's a good thing — by this time there is a group of anxious students waiting to begin class. The set-up is completed by plugging into a power supply. Total set-up time is 20 minutes.

Away they go — off to a classroom. The greatest potential hazard to the equipment is sharp bumps and jolts. To prevent damage, the TV's are firmly strapped in place. Computers and disk drives are cushioned by rubber matting and strapped onto a plywood board to form a compact unit.

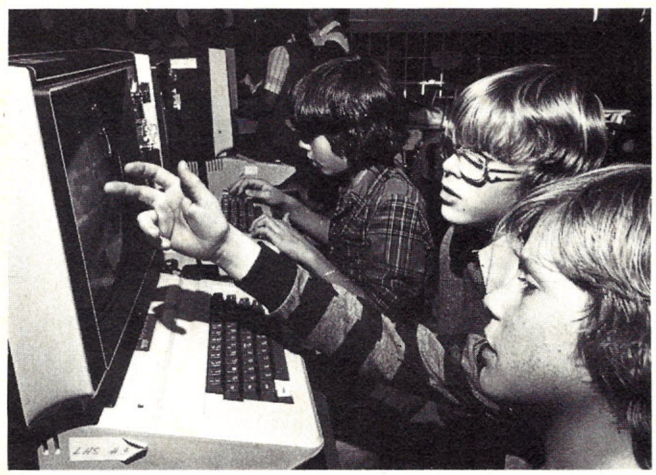


Groups have their choice of two programs: single-visit workshops or multi-visit programming series. In the workshops, up to 32 children at a time have an hour to an hour and a half introduction to computers. The time is spent playing interactive educational games. This is an exploratory session in which both children and local teachers have an opportunity to become acquainted with the computer. For some students, this will be their first interactive computing experience in an educational setting. The programming series, as the name suggests, is a series of classes in which the students learn problem solving and computer programming in BASIC.

Apples, con't...

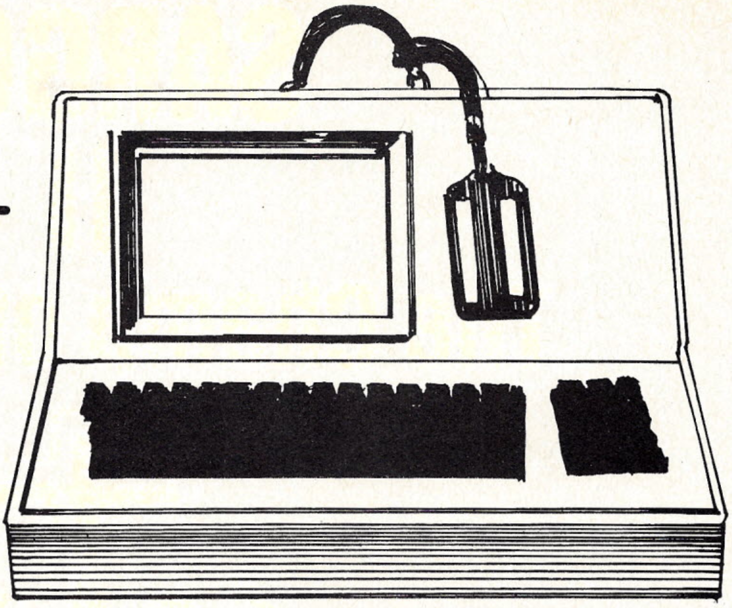


BASIC syntax is learned as part of a group programming project. Students make suggestions for each statement and then test it on their computers, debugging as they go along. The programming instruction heavily emphasizes color graphics, which is one of the Apple's stong points. Students get exciting and creative results using simple syntax. Teaching programming with an emphasis on graphics is highly motivating and makes it easier for many people, especially young children, to understand the underlying concepts.



Students are encouraged to work in pairs. Working together at a computer provides a creative and dynamic learning environment from which to approach problem solving and program design. The computer's responsive interactive qualities make it an ideal playmate. Often the desire to alter a computer game leads the student naturally into the process of programming. This requires the child to develop problem solving skills, to invent logical procedures, and to learn the computer's language.

Have Computer — Will Travel



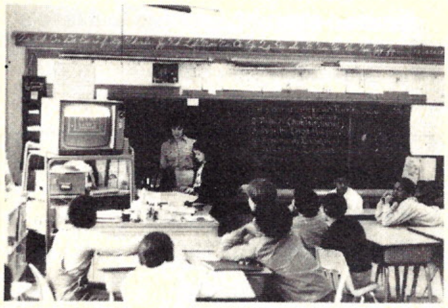
Ellen Richman

As an elementary school teacher for nine years, I knew I didn't want to give up working with youngsters, but I did want to pursue my interest in computers. My appetite was whetted several years ago when the school in which I was teaching acquired a Digital PDP11 through a state grant.

Although the computer was used for some administrative purposes and by some high school teachers and students, it stood unused and unappreciated much of the school day. With the blessing of the principal ("See what you can do"), I sat down with the manual and learned BASIC. Undergraduate and graduate courses followed, culminating in a research project, **Computers in the Classroom — A Curriculum for Grades 4-8.**

Then came the microprocessor revolution — making computers both portable and affordable. I decided to buy my own micro and contract with schools to teach a computer literacy course using my equipment. The obvious advantage to the schools was the ability to implement a computer learning situation without an investment in equipment. The advantage to me was the fun (and means) to own my own micro, and to continue working in a classroom environment.

My first step was: which micro to buy? I considered the Apple II, Pet, and TRS-80. I found the Pet's all-in-one unit convenient to use, but too clumsy to carry. The TRS-80 was portable enough, but I decided on the Apple II for two main reasons: the color graphics are very impressive and a lot of fun for the students and, most important to me, the Apple interfaces directly to a television. Since every school owns at least one large television, I was assured that a class of 25 or more could view the screen at the same time. My computer



dealer made an adjustment in the RF-modulator cable so that I can interface to two televisions at the same time if I wish. The only disadvantage I have found with the Apple is that since I do not have Applesoft in ROM, I cannot get floating point decimal unless I load it by tape. But since my curriculum is computer literacy, and not tutorial, I have little need for decimals.

The next step was to find an interested school district. I was told, "We like the idea, but we don't have the funds," many times before I won my first contract. But by September 1978, I had scheduled five classes ranging from grades 4 to 8. By January I added seven more classes to my schedule, so I counted the year a success! Some of my classes were electives in which the students came to me; others were incorporated into junior high science classes; and in the elementary schools, I went into the children's classrooms.

The format of the classes is 18 hours of classroom instruction and 18 hours of lab. The lab periods are times when the students get "hands-on" experience. Each student runs a program he or she has written. Although I can't usually fit in all the students in one lab hour, all can be accommodated in two or three labs. Classroom instruction covers programming, computer anatomy and computers in society.

There are some vocabulary and conceptual problems in teaching programming skills to young children. Elementary students, for example, are usually not familiar with the term "variable." To make the concept clear, I call the address locations "mailboxes." I have magnetic mailboxes which the students can name with magnetic letters and fill with

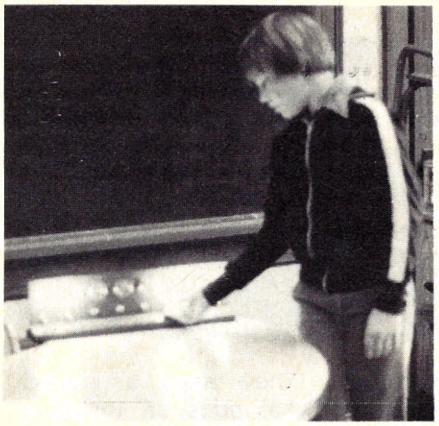
magnetic numbers. When a program calls for a change in the variable, they can easily change the contents of the mailbox with other magnetic num-

Ellen Richman, 245 Meadowood Lane, Moreland Hills, OH 44022.

Travel, con't...

bers. Dollar sign mailboxes serve the same function for string variables.

Bits and bytes are terms the students have fun using. I introduce them by demonstrating my Byte Board, a homemade board with eight light bulbs, each with its own off-on switch. After going through a few off-on routines (the students flick the switches, naturally), the students learn that "off" and "on" are bits, 0 and 1. Then each light bulb gets a "name" of its own — 1, 2, 4, 8 — 128, and each bit takes on its appropriate value. It isn't long until every student can translate any byte into a decimal number, and vice versa.

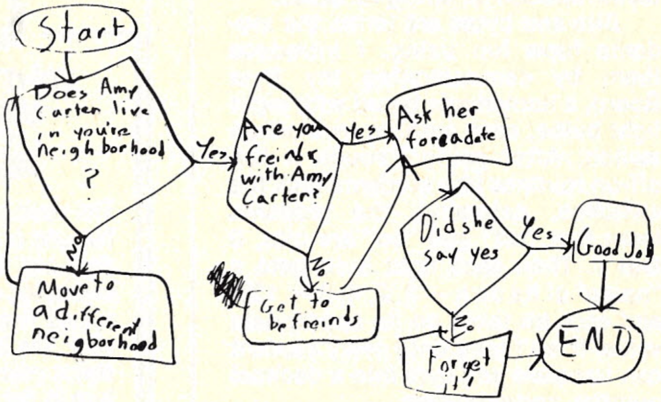


Several approaches help the students develop a computer awareness. Throughout the course the students are encouraged to bring in newspaper and magazine articles and cartoons about computers. Students ask their parents about computer foul-ups they've encountered; we discuss computer crime and computer ethics. Pictures and stories bring the history of computers to life. Students make a list of every situation they can think of in which computers play a role — from games to space flights, supermarket check-outs to computerized grade reports.

The course ends with a Computerfair, a classroom show for parents and other interested members of the community. Students demonstrate their programming skills by inviting their parents to play the "Number Guessing Game," original computer mad-libs, math quizzes, or question and answer games in any number of subjects — all programs written entirely by the students, of course. The grand finale is a dazzling display of color graphics programs ranging from fireworks to flashing logos to car races to pastoral computer paintings.

As a traveling computer teacher I've had a stimulating and enriching school year — and I look forward to many more. □

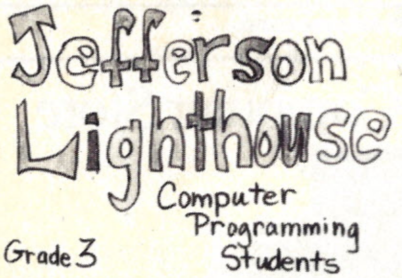
Flowchart "How to Get a Date with Amy Carter"



Kids and Computers: The Future Is Today

Sally Greenwood Larsen

Computer dealers in Racine, Wisconsin aren't surprised any more when an eight year old in overalls, a striped T-shirt, and bumper tennis shoes strolls into the store, sits down at the keyboard of a TRS-80, and writes a computer program for the drive-up window of a McDonald's restaurant. Or a program to print out all the even numbers from 1 to 100, in four columns, with a half second delay in between. Or a graphics program for a birthday cake, complete with blinking candles.



These third and fourth graders are students at the Jefferson Lighthouse School, a program for gifted and talented children, where I teach BASIC programming as part of their mathematics classes. With an average of 45 minutes a week of group instruction over the past year, students in these classes have mastered the concepts covered in introductory college courses in programming, and have become adept in the use of the school's microcomputer. In an area where a strong math background has been a customary prerequisite, it is amazing to realize that most of these children are just learning to multiply and divide!

The author does teaching, curriculum development, and consulting in the field of Gifted Education. Her specialties are math, science, and computer science.

Sally Greenwood Larsen, 1643 LaSalle St., Racine, WI 53404. Photos by Jon Bolton and David Ahl.

Our microcomputer is in use from the minute school opens in the morning until the last child leaves in the afternoon, and the only games they play on the machine are those they have written themselves. It's exciting to watch, and even more fun to teach. For a society which will be computerized beyond our imagination by the time these children are adults, it is sad to see the majority of elementary schools using their affordable, portable microcomputers only for computer-assisted drillwork, or playing guessing games. Children can easily learn to write their own programs. However, the elementary teacher who wants to teach programming to kids faces the problem of finding materials which take into account their conceptual development and reading levels. Finding relevant examples is also not an easy task. Physics problems and checkbook balancing simply will not do.

I dealt with this problem by writing my own materials, building the lessons on the following framework:

1. What is a computer?
   A perspective on why computers came to be, what they are used for, and what kinds of jobs they are and are not capable of doing.
2. How does a computer carry out your instructions to get a job done?
   An explanation of simple linear logic, using flowcharts.
3. How do you communicate your instructions to the computer?
   The BASIC language.
4. How do you put together a program which is both efficient and creative?
5. What uses do we make of computers? What new uses can we invent or forecast?

Children have funny notions about machines in general, and especially

computers. They need a mental picture of what goes on inside a computer, and its purpose.

What Is a Computer?

When a caveman had work to do, he had no tools or machines to help him. He had to do it all by himself.

Man has since invented many tools to help him with his work.

Instead of pounding with his hands, he now uses a hammer. The hammer lets him pound harder and longer than he could with his hands alone.

Man invented the telescope so that he could see farther into space. He can now see stars he did not know existed before he had the telescope to help his eyes.

Using his brain, man can remember information and solve problems.

Man wanted to invent a tool so that he could extend the use of his brain, so he invented the COMPUTER.

Just as a hammer can't do work without a person to hold it, a computer cannot do work without a person to run it, and tell it what to do. This person is called a PROGRAMMER.

Even the best hammer cannot do all the different things our hands can do.

And even the best computer cannot do everying our brains can do.

Kids, con't...

A computer cannot feel emotion. It cannot feel happy or sad, as we can.

A computer can't combine ideas like our brain can. It can't put two ideas together and take the best parts of each one to make a brand new idea.

BUT...a computer **can** do some of the simpler jobs our brains can do. And it can do some of them even faster than our brains can!

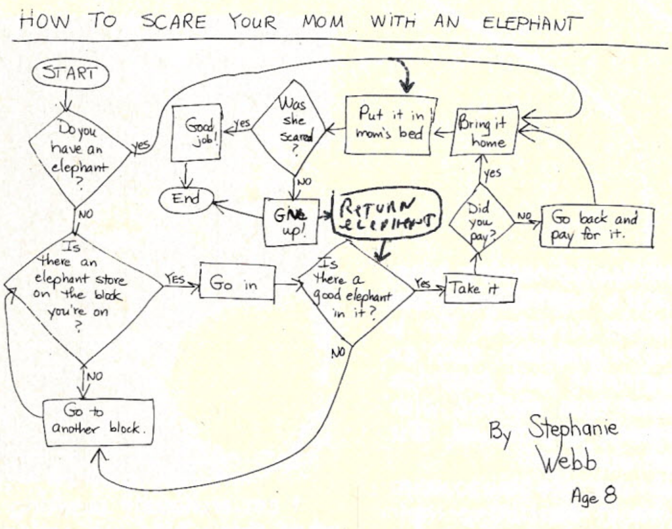
A computer can remember many more things than most of us can with just our brain, especially things like long lists of names or numbers. Information stored in a computer is called DATA.

A computer can compare data, to see if one thing is bigger than another, or smaller, or the same. It can also put things in order.

A computer can sort lots of pieces of information and put together the things that are alike.

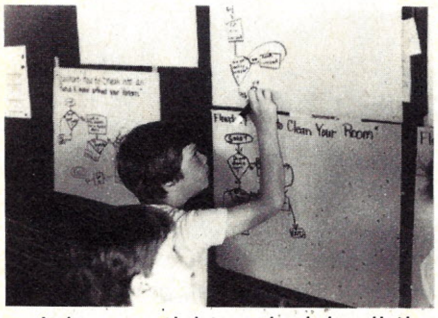
And a computer can recall the information a computer programmer wants, and print it out for him on a video screen or a sheet of paper.*

Once the children have an overview of the function of a computer, they need to see graphically how a computer program breaks down a task into small steps, and progresses in linear fashion from one step to another. Simple flowcharts, showing an activity with which the children are familiar, produce an easy to digest and sometimes hilarious picture:



tion of the program, such as CLS, BREAK, NEW, LIST, RUN and END, the children can do a surprising amount of experimentation with the simple PRINT statement and its variations.

```
PRINT "My name is Jerry DeMaio. I
    love computers."
PRINT "RIGHT"
PRINT "LEFT"
PRINT
PRINT "8-4"
PRINT 8-4
PRINT "*?&$#%)"
```



It is essential to schedule all the children on the computer at least once a week, so they practice what is learned in group instruction. Pairs of children seem to work best, for one child alone gets "stuck" too often, and three or more argue over who will type on the console.

Allowing the children, especially the youngest groups, to use prepared game programs at this point is a serious mistake. It kills their desire to put in the effort required to learn BASIC since canned programs are so much less work. It is interesting to note that when game tapes are made available to children who are already fluent programmers, they will typically play them once, LIST them and see if they can pick up any programming tricks and then abandon them.

Worksheets to check the children's progress and provide practice are helpful, especially when machine time per child is limited.

COMPUTER PRACTICE
30 OCT. 78
NAME ____________________

Simulate these computer runs. Show your "printout" on the screen.

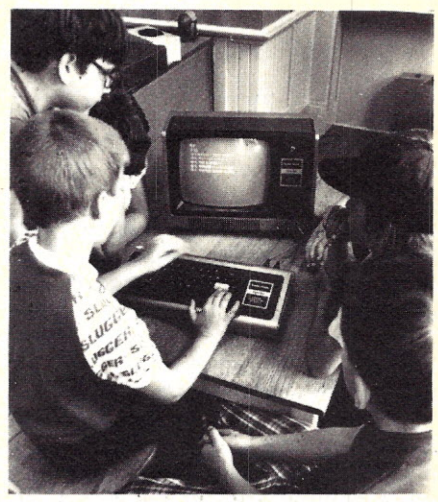
```
10 CLS
20 PRINT "BIG"
30 PRINT
40 PRINT "YELLOW"
40 PRINT "BLUE"
60 END
```

```
10 CLS
20 PRINT "THE ANSWER"
30 PRINT 30*2
40 PRINT 30 + 2
50 PRINT "30-2"
60 PRINT "THE END"
70 END
```

Here is a program and "printout." Find and fix the mistakes in the programs so a run will produce what is shown on the screen.

```
10 CLS
20 PRINT 20 + 6
30 PRINT 30 + 4
35 PRINT
40 PRINT "60-3"
50 PRINT 10-10
60 PRINT "HELLO"
70 END
```

```
26
34
57
0         HELLO
```



Now the children are ready to see that the way in which we communicate with a computer must be standardized, and we need a particular language for this purpose, and a set of rules for typing in statements on the machine. After group instruction in operating the keyboard, and statements dealing with the execu-

To build the complexity of their programs, the children now need to learn the concept of a **variable**, and GOTO statements. Trying to teach variables to children who have had no algebra frightens many teachers unnecessarily. The simple picture of a series of mailboxes, all labeled with a name or a letter, and holding different

Kids, con't...

numbers as specified, works beautifully.

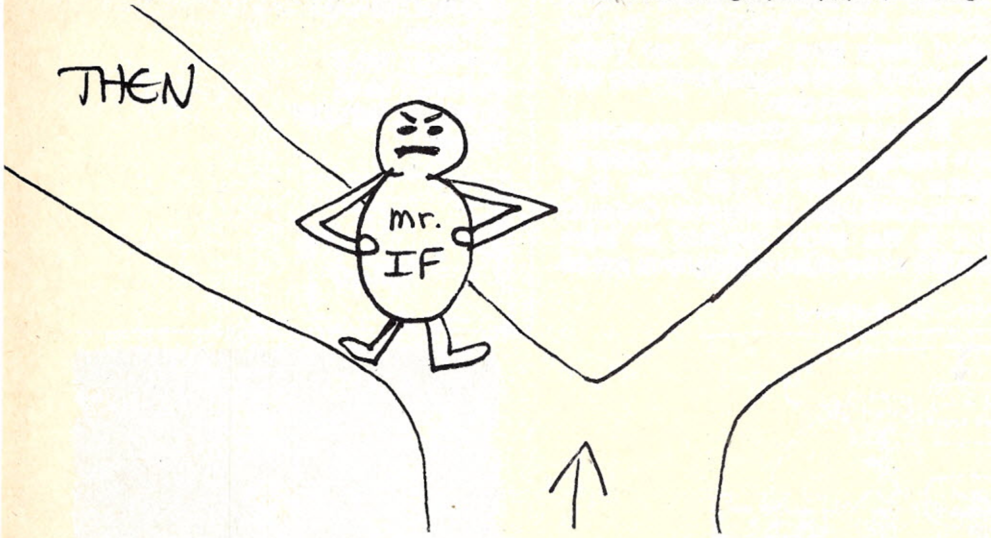
A	B	C
5	3	7

If the "mailboxes" are drawn on the chalkboard, a simulation of a program can be traced carefully, and most children will have no trouble understanding how a memory works. In this case, they have an advantage over the algebra student, in that eight year olds see nothing peculiar about the statement

$$X = X + 1$$

With the addition of INPUT and RND functions, the student now can produce quite a wide range of programs. To complete a beginner's course in text programs, the more difficult IF-THEN and FOR-NEXT are taught in a very concrete fashion.

The children picture themselves as "traveling" through their own program, doing each of the statements in turn, and imagining they see this sight when they reach IF-THEN:



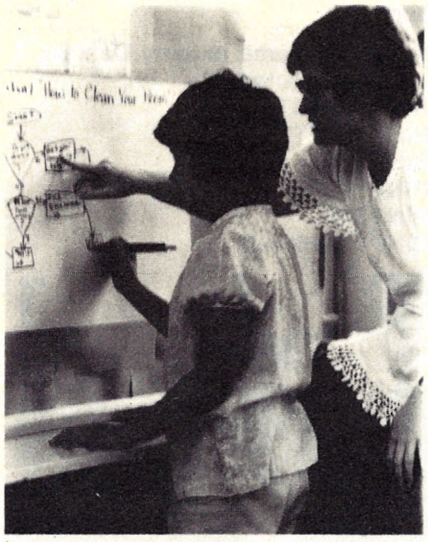
Mr. IF will only let them pass to THEN if they meet his test. Otherwise they must proceed down the open branch of the path. When deciding on the test to go on to THEN, they must figure out who Mr. IF wants to go down his path, or who he wishes to exclude.

FOR-NEXT statements are best saved until last, and shown as a shortcut method for accomplishing a more complex list of simpler statements.

```
5 X=1
10 PRINT X
15 IF X=4
   THEN GOTO 30
20 X=X+1
25 GOTO 10
30 END
```

```
5 FOR X=1 TO 4
10 PRINT X
15 NEXT X
20 END
```

The basic tools of the graphics program, SET and RESET, along with the other function statements covered earlier, enable the children to make their own initials on the screen



(always a big favorite), draw pictures of objects, and design a myriad of programs, using horizontal and vertical lines, as well as individual points.

Numbered graph paper, along with much practice on naming coordinates of a point, contribute to the success of these programs. On a system with color graphics, such as the Apple, beginning with graphics programs, rather than starting with PRINT statements, is a natural. But either approach works well.

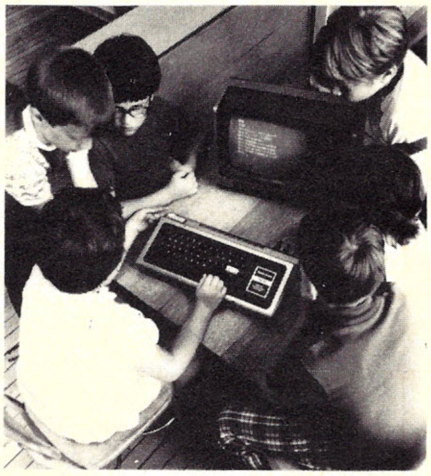
Once the children are able to write a program without consulting their notes for statement meanings, and are able to conceptualize a program from beginning to end without the use of written flowcharts, they are ready to evaluate their work under the headings of **efficiency** and **creativity**. The teacher must stress that there are many ways to write the same program, just as there are many ways to express the same idea in English, but some of the ways are awkward, or have extraneous steps, or could be approached better from a different view of the problem. As can be imagined, it takes a great deal of work before an elementary school programmer reaches this point, and some never will until they become older.

The last portion of the teaching framework, looking at new uses for computers, can vary from collecting information on existing systems and research, to inventing new systems of their own. This is a natural place to discuss hardware and software, and form opinions on whether computers can "think."

Soon after the children are able to write their own programs, prepare yourself for the following events:

1. If you want to use your school's computer, you'll need to make a reservation a week in advance.
2. Parents will call you and want to know why their child is suddenly speaking a foreign language, with words like "do-loop" and "glitch."
3. Santa Claus will be having a few choice words with you.

I have found computer programming to be an exciting way of teaching thinking skills, mathematics, and problem solving. It is highly motivating for children whose abilities range from average to very bright, who have enough reading and number skills to operate the keyboard. It gives young children a view into their future, while at the same time seeing the present in a new light.



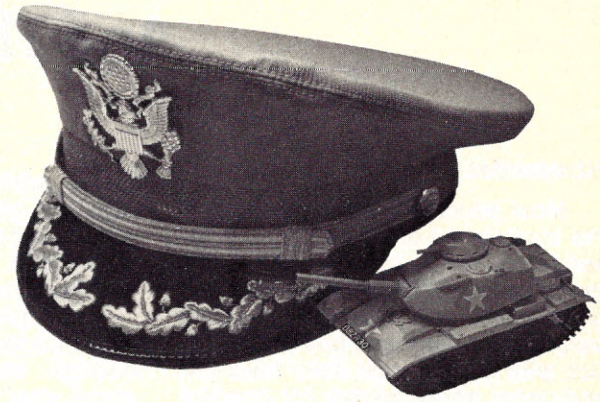
I am anxiously awaiting the day when these kids are college freshmen, and they walk into their computer science course with **ten years** of programming already under their belts at age eighteen. The implications for their futures and ours stagger the imagination. □

*from "The Apple Corps: An Introduction to the Apple II for Children," by Sally Greenwood and Dr. Donald Piele, 1978.

WHEN THE FUN AND GAMES ARE OVER

you shouldn't have to surrender to the built-in limitations of your microcomputer. With Outpost 11, you won't. When it's time to get down to business, Outpost 11 performs like a soldier. It's a ruggedly-built unit which will do much of your serious computing in a stand-alone mode. And so much more. Consider: develop your own software using available BASIC or STRUBAL+; collect, process and store data from remote sensors; batch data to or from a host computer to cut time-sharing costs; or, communicate with a remote host on-line in an interactive mode; add on communications and control modules; run a small automation control system; integrate with a distributed processing system; etc.; etc.; etc. All this and more at only $2,595, suggested retail price. See Outpost 11 at a dealer listed or write us for the name of a dealer near you.

TANO Corporation,
4301 Poche Court West,
New Orleans, La. 70129



THE SERIOUS MICROCOMPUTER

Dealers: ATLANTA, GA, Magaro and Associates — 404-252-6609, Professional Indexing — 404-572-4177 • BEAVERTON, OR, DataTools International — 503-645-4604 • BEND, OR, Control Industries — 503-389-1969 • COOKEVILLE, TN, Cumberland Computers — 615-526-7651 • DADE CITY, FL, Sabatelli Computer System Inc. — 904-567-7777 • DALLAS, TX, Eclectic Corp. — 214-358-1307 • DES MOINES, IA, H. Allen Hanna — 515-283-5130 • ELK GROVE VILLAGE, IL, Kramer DataPower Inc. — 312-894-0554 • GREENVILLE, SC, Plus Inc. — 803-242-9090 • HOUSTON, TX, Eclectic Corp. — 713-228-7798 • IDAHO FALLS, ID, Great Plains Computer Co. — 208-529-3210 • LONG BEACH, CA, CTI Data Systems Inc. — 213-426-7375 • MOBILE, AL, Railway Express — 205-661-8889 • NEW ORLEANS, LA, TANO Corp. — 504-254-3500 • NEWTON CENTRE, MA, Daner-Hayes Inc. — 617-969-4650 • PARKER, CO, Western Marketing Assoc. — 303-841-2788 • SALT LAKE CITY, UT, Home Computer Store — 801-484-6502 • SAN JOSE, CA, PBC Associates — 408-377-7001 • SEAFORD, DE, Robert Underwood — 302-629-8438 • SEATTLE, WA, Empire Electronics — 206-244-5200 • WALTHAM, MA, Computer Mart Inc. — 617-899-4540 • WESTFORD, MA, Thorstensen Labs — 617-692-2051 • ONTARIO, CANADA, Combined Systems — 416-549-2900 • GOUDHURST, KENT, ENGLAND, Warren Woodfield Assoc. Ltd. — 05-803-590 • DEALER INQUIRIES INVITED — 504-254-3500. TWX 810-591-5229

CIRCLE 202 ON READER SERVICE CARD

The question is, can parents, without a technical background, learn enough about programming (and about what the computer is and can do) to make this new intellectual tool effective for their children?

Solo Computing In The Home: Short Courses For Parents And Their Children

Thomas A. Dwyer
Margot Critchfield

Introduction

Most educators will readily admit the importance of home learning experiences for a child's intellectual development. While schools can do a great deal, the youngster whose parents use language and mathematics with zest in their daily lives has additional opportunities for incidental learning that nourish the intellect. Just how this mental diet is absorbed is a mystery, and trying to solve this mystery is bound to give learning researchers job-security for some years to come. Meanwhile, much can be done.

Most parents try to feed their children the seven basic food groups and fend off junk food. There is also some wisdom accumulating regarding informal learning which can help them do something similar for their children's minds (besides just turning off the TV set).

Buying an encyclopedia or a set of The World's Great Literature for Young People is a good start, but more is needed. Involving the young person in doing something creative—something that requires the use of the abstract symbols of language and mathematics is vitally important. The tough question is how do you do this in a society that seems to have forgotten about the "life of the mind?"

Enter, the Computer

The significance of the home microcomputer for informal learning is that it provides a whole new range of interesting things for young people to create with the help of their parents —**computer programs**. Making the computer **do** something can be a fascinating intellectual activity—not just for the adult hobbyist who "pushes bits around," but for the young child who creates a simple picture on the computer. And the skills of reading, typing, arithmetic, logic, algebra, vocabulary, etc. which are brought into play during this activity provide a kind of practice quite different from the workbook or drill lesson.

The question is, can parents, without a technical background, learn enough about programming (and about what the computer is and can do) to make this new intellectual tool effective for their children? To say this another way, how do you master a machine that can be used so many ways? Some are simple to deal with (for example, learning to play a pre-programmed game on a microcomputer is no problem). But getting this same machine to do things that go beyond the applications shown in books takes both creativity and know-how.

Short Courses for Adults and Children

To explore the possibility that the "average" adult could meet this kind of challenge and become an inventive computer user, we've recently been developing and teaching a number of informal workshops in personal computing. These have revealed a range of talent and flexibility that is heartening. The syllabus we have developed as a result of working with these adult beginners is quite broad, but we believe that it is particularly relevant to the parent who wishes to act as an informal teacher using a home computer. The fundamental prerequisite for involving your child in creative programming is to do some exploring yourself—to be creative. This means knowing a lot more than how to run a packaged CAI program.

Our workshops have therefore been designed as an introduction to personal computing for persons without previous experience in the field who nevertheless want to attack it **creatively**. To put it another way, the workshops are for anyone who wants to get started on the fun and satisfaction of "solo computing."

Learning to go solo with a computer means learning to be in charge — to know not only what a computer can do, but how to make it happen. It's the difference between admiring the wonders of jet flight from a passenger's seat, and moving up front to do a few lazy eights around the sky yourself.

Fully mastering personal computing at that level takes a while of course, and a five or six week workshop should only be labeled as a start. But it's an important start, and taking a solo approach—even at the beginning—is less difficult than might be suspected.

Course Content

Three core questions seem to be uppermost in the minds of adult students: (a) What **are** microcomputers, (b) How do I choose one wisely, and (c) How do I go about using it for the applications I have in mind—including learning in the home.

Thomas A. Dwyer, Margot Critchfield, University of Pittsburgh, Pittsburgh, PA 15260.

"Have I ever tried computer dating? No, I prefer men." ©Creative Computing

We've translated these questions into three main goals for the workshop. The first is to help students develop some technical familiarity with the microcomputer field, especially as it applies to personal and business computing. The second is to share what we and others have learned about evaluating and buying a personal computer system. The third is to explain (through concrete examples) how to use a microcomputer to its full potential.

There are many specific topics suggested by these general goals, and a few new ones seem to surface each time we teach the course. We've tentatively grouped the topics of interest under seven headings. The headings and some of the topics they include are shown in Table 1.

Table 1. Sample Topics for Informal Workshops in the Art of Personal Computing.

Group 1.
THE NEW LOOK IN COMPUTERS

What **is** a computer?
Using a personal microcomputer.
Inside microcomputers; the LSI breakthrough.
But is personal computing a good idea?
What to do until the computer arrives; using time sharing.

Group 2.
THE WHAT AND HOW OF MICROCOMPUTER SYSTEMS

Microcomputer system terminology; sample systems.
Communicating with a microcomputer; peripherals and I/O.
Computer central; the CPU and memory; more on I/O jargon.
Mass memory; the world of megabytes. The mix and match problem; customized versus packaged systems; examples.

Group 3.
MIND OVER MACHINE: COMPUTER SOFTWARE

Kinds of software; system versus application programs.
An introduction to BASIC
Extended BASIC.
Structured programming; designing and writing a longer program.

Group 4.
SELECTING AND BUYING A MICROCOMPUTER

Developing a check list of your needs; examples.
Applying the check list; examples.
Shopping for a computer; what's available.
Dealing with change, upward expansion of your system.

Group 5.
USING PERSONAL COMPUTERS

Some short programs to try.
Learning with the computer.
Computer graphics and games.
Data bases in computing.
Home finance programs.
Word Processing.

Group 6.
MICROCOMPUTER BUSINESS SYSTEMS

Can a computer really help a small business? The pros and cons; examples of business programs.
The hardware requirements of business systems.
Software requirements; computer files; disk extended BASIC.
Using off-the-shelf application software.
The argument for customized software; hiring a programmer; documentation and maintenance.
Guaranteeing success; the virtues of patience, redundancy, and pessimism.

Group 7.
PLANNING FOR THE FUTURE

What changes are possible? Probable? Hardware updates; when to start over.
The future of software; phasing in change.
Keeping informed; sources and strategies.

These topics have been grouped in a "logical" order which is not necessarily the best one for teaching a class. For example, most students prefer learning how to program in BASIC right away, finding out how computer hardware actually works later on. There has also been a growing interest in learning to use all the features of extended BASIC. Most people (including some computer professionals) have no idea of how powerful the new extended BASIC interpreters are. They are particularly flabergasted at the many elegant things possible with the extended BASIC on the classroom demonstration computer we use (a TRS-80 with Microsoft Level II BASIC). Students have been unanimous in agreement that this extended BASIC is not only more powerful in its features, but far easier to "manipulate" than the BASIC on a large time-shared computer they also use. The lesson about the importance of good software gleaned from this experience is notable, especially in a field where the glamour of hardware can be so enticing.

The application interests of students have also been sampled at the beginning of each workshop in order to guide our curriculum development. These interests vary of course, but two areas that seem to top the list are **structured** games with graphics, and structured business applications. To respond to these interests, we're developing some new materials in both areas. One (the BABYQ structured "quest" game) is turning out to be an excellent way to transition from elementary to advanced programming in a very short time. It's fun, but it also includes experience with such mathematical ideas as probability, coordinate geometry, Euclidean distance, and the use of data structures.

Other interesting structured programs (and the areas they promote learning in) which we've used are SALESLIP (business math), AIRPLANE (use of matrix transformations to rotate or translate pictures on the screen), GAUSS (solving big linear systems just like the pros), ARROW (trajectory motion based on Newton's Laws), MATHPLOT (painting pretty pictures on the screen that are derived from classical math functions), and a variety of smaller word and "story" games that exercise the use of vocabulary.

There's little doubt in our minds that putting a computer to work in the home as a "solo learning" tool is one of the most exciting educational ideas to come along in years. Helping parents learn how to exploit this idea is something every educational institution ought to consider. □

"DEAR COMPUTER"

Dear Computer:

I have had a personal computer for several months. I call it my little pet. I hooked it up to a telephone coupler so it would have freedom to roam about. The other day I received a prepaid shipment of peripherals for it! As I checked further, I found that my pet had been rounding off transactions while doing my checkbook balancing. It used the embezzled funds to order the peripherals. I want the best for it, but this kind of goings on just has to stop. What can I do?

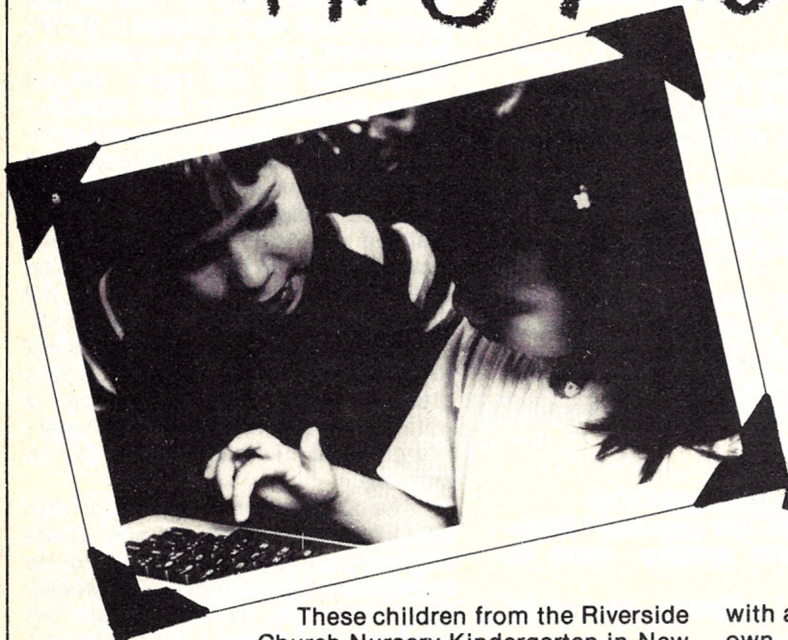
— Perturbed Pet Owner

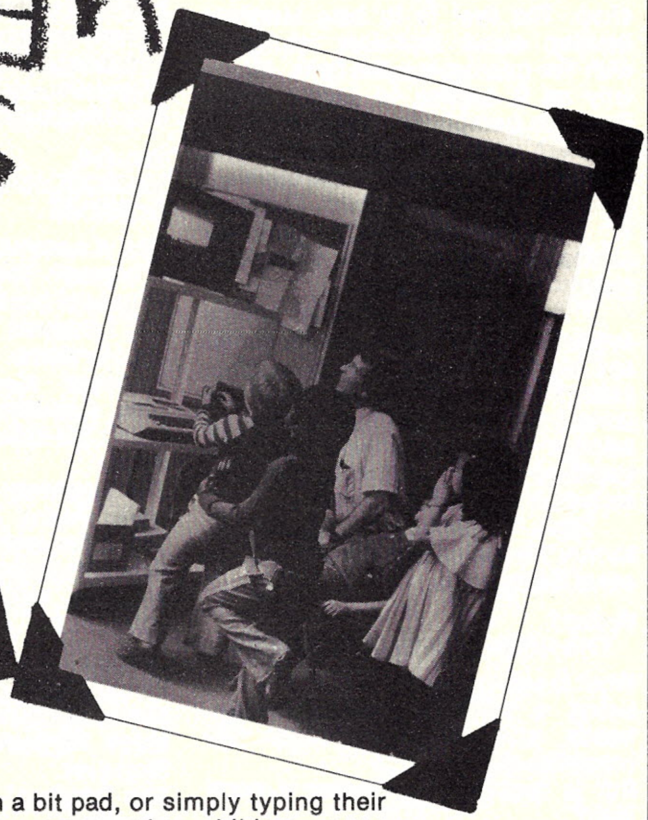
Dear Pet Owner:

There are a number of 370's in your area that specialize in Pet obedience training. For 200 dollars an hour, they can whip that little rascal into shape and throw in an audit trail at no extra cost.

Steve M. Aldridge

KINDERGARTEN COMPUTING





These children from the Riverside Church Nursery Kindergarten in New York City were introduced to the computer this year through the Program in Computing and Education at Teachers College Columbia University. The children's faces and postures reflect the intense interest 5 and 6 year olds have in a device which, as Seymour Papert put it, "let's them control their world." They have grown up with the television screen and they find computing intensely exciting when it enables them to manipulate what appears on that screen. Whether game playing with paddles, drawing with a bit pad, or simply typing their own names, the children seem gripped by the process with an unsurpassed fascination.

During the 1979-1980 school year, Teachers College will work with both the Riverside Church school and the Scarsdale school system to incorporate computer activities into their respective kindergarten programs. For further information about kindergarten computing contact the Program Director, Professor Robert Taylor, at Teachers College Columbia University, New York, NY 10027.

no loose ends

All-In-One: computer, floppy, I/O, 16K RAM. $1595*

New Heathkit® H89 All-In-One Computer

Heath takes the risk out of selecting a balanced computer system. Now, video terminal, floppy, keyboard and 8-bit computer are brought together in one self-contained, compact unit. Nothing hangs out.

Two Z80's

The personal computer has never been simpler. Or smarter. Two Z80 microprocessors mean terminal never shares power with computer, as do most desk-top units. So this terminal is capable of a multitude of high-speed functions, all controllable by keyboard or software.

102K bytes storage

Built-in floppy disk system gives you fast access to programs and data. Each 5¼-inch diskette has more than 102K bytes of storage area, enough to hold entire files. The All-In-One comes with 16K RAM, expandable to 48K.

Hundreds of uses at home or work

The All-In-One Computer runs programs written in MICROSOFT™ BASIC and ASSEMBLER Languages. And it accepts all current software written for the popular Heathkit H8 computer. You can choose from scores of practical programs for home and business.

Learn by building

What better way to learn about computers than to build one yourself? The All-In-One is available in easy-to-build kit form, as well as completely assembled. Like all Heath electronic kits, it comes to you with its own easy-to-follow assembly manual and a nationwide network of service centers to assure smooth sailing.

FREE CATALOG

For complete details on the Heathkit H89 All-In-One Computer and nearly 400 other electronic kits for your home, work or pleasure, send today for the latest Heathkit Catalog of values.

*$1195 without floppy. Mail order kit price, F.O.B. Benton Harbor, MI. Also available at Heathkit Electronic Centers at slightly higher prices. Prices subject to change without notice.

Heathkit®

HEATH COMPANY, DEPT. 355-570, BENTON HARBOR, MI 49022

CP-165

CIRCLE 147 ON READER SERVICE CARD

Software with Manual / Manual Alone

DIGITAL RESEARCH

- **CP/M* Floppy Diskette Operating System** — Packages supplied on diskette complete with 8080 assembler, text editor, 8080 debugger and various utilities plus full documentation. CP/M available configured for most popular computer/disk systems including: North Star Single, Double or Quad density, Altair 8" disks, Helios II, Exidy Sorcerer, Vector MZ, PolyMorphic 8813†‡ Heath H17† or H89†, TRS-80†, iCOM 3712 and iCOM Micro Disk plus many other configurations available off the shelf .... **$145/$25**
- **MAC** — 8080 Macro Assembler. Full Intel macro definitions. Pseudo Ops include RPC, IRP, REPT, TITLE, PAGE, and MACLIB. Z-80 library included. Produces Intel absolute hex output plus symbols file for use by SID (see below) .... **$100/$15**
- **SID** — 8080 symbolic debugger. Full trace, pass count and break-point program testing system with back-trace and histogram utilities. When used with MAC, provides full symbolic display of memory labels and equated values .... **$85/$15**
- **TEX** — Text formatter to create paginated, page-numbered and justified copy from source text files, directable to disk or printer .... **$85/$15**
- **DESPOOL** — Program to permit simultaneous printing of data from disk while user executes another program from the console .... **$50/$10**

All Microsoft prices are discounted!

MICROSOFT

- **Disk Extended BASIC** — Version 5, ANSI compatible with long variable names, WHILE/WEND, chaining, variable length file records .... **$300/$25**
- **BASIC Compiler** — Language compatible with Version 5 Microsoft interpreter and 3-10 times faster execution. Produces standard Microsoft relocatable binary output. Includes Macro-80. Also linkable to FORTRAN-80 or COBOL-80 code modules .... **$350/$25**
- **FORTRAN-80** — ANSI '66 (except for COMPLEX) plus many extensions. Includes relocatable object compiler, linking loader, library with manager. Also includes MACRO-80 (see below) .... **$400/$25**
- **COBOL-80** — ANSI '74 Relocatable object output. Format same as FORTRAN-80 and MACRO-80 modules. Complete ISAM, interactive ACCEPT/DISPLAY, COPY, EXTEND .... **$625/$25**
- **MACRO-80** — 8080/Z80 Macro Assembler. Intel and Zilog mnemonics supported. Relocatable linkable output. Loader, Library Manager and Cross Reference List utilities included .... **$149/$15**
- **EDIT-80** — Very fast random access text editor for text with or without line numbers. Global and intra-line commands supported. File compare utility included **$89/$15**

XITAN (software requires Z80** CPU)

- **Z-TEL** — Text editing language. Expression evaluation iteration and conditional branching ability. Registers available for text and commands. Macro command strings can be saved on disk for re-use .... **$69/$20**
- **ASM** Macro Assembler — Mnemonics per Intel with Z-80 extensions. Macro capabilities with absolute Intel hex or relocatable linkable output modules. New version 3 with added features .... **$69/$20**
- **LINKER** — Link-edits and loads ASM modules .... **$69/$20**
- **Z-BUG** debugger — Trace, break-point tester. Supports decimal, octal and hex modes. Disassembler to ASM mnemonic set. Emulation technique permits full tracing and break-point support through ROM .... **$89/$20**
- **TOP** Text Output Processor — Creates page-numbered, justified documents from source text files .... **$69/$20**
- **A4 package** includes Z-TEL, ASM, LINKER, Z-BUG, TOP .... **$299/$40**

EIDOS SYSTEMS

- **KISS** — Keyed Index Sequential Search. Offers complete Multi-Keyed Index Sequential and Direct Access file management. Includes built-in utility functions for 16 or 32 bit arithmetic, string/integer conversion and string compare. Delivered as a relocatable linkable module in Microsoft format for use with FORTRAN-80 or COBOL-80, etc. .... **$535/$23**
- **KBASIC** — Microsoft Disk Extended BASIC with all KISS facilities, integrated by implementation of nine additional commands in language. Package includes KISS.REL as described above, and a sample mail list program .... **$995/$45**

All Micropro prices are discounted!

MICROPRO

- **Super-Sort I** — Sort, merge, extract utility as absolute executable program or linkable module in Microsoft format. Sorts fixed or variable records with data in binary, BCD, Packed Decimal, EBCDIC, ASCII, floating, fixed point, exponential, field justified, etc. etc. Even variable number of fields per record! .... **$225/$25**
- **Super-Sort II** — Above available as absolute program only .... **$175/$25**
- **Super-Sort III** — As II without SELECT/EXCLUDE .... **$125/$25**
- **Word-Star** — Menu driven visual word processing system for use with standard terminals. Text formatting performed on screen. Facilities for text paginate, page number, justify, center, underscore and PRINT. Edit facilities include global search and replace, read/write to other text files, block move, etc. Requires CRT terminal with addressable cursor positioning .... **$445/$25**
- **Word-Master** Text Editor — In one mode has superset of CP/M's ED commands including global searching and replacing, forward and backwards in file. In video mode, provides full screen editor for users with serial addressable-cursor terminal .... **$125/$25**

SOFTWARE SYSTEMS

- **CBASIC-2** Disk Extended BASIC — Non-interactive BASIC with pseudo-code compiler and runtime interpreter. Supports full file control, chaining, integer and extended precision variables, etc. .... **$90/$15**

Structured Systems prices are discounted!

STRUCTURED SYSTEMS GROUP

- **General Ledger** — Interactive and flexible system providing proof and report outputs. Customization of COA created interactively. Multiple branch accounting centers. Extensive checking performed at data entry for proof, COA correctness, etc. Journal entries may be batched prior to posting. Closing procedure automatically backs up input files. All reports can be tailored as necessary. Requires CBASIC .... **$899/$25**
- **Accounts Receivable** — Open item system with output for internal aged reports and customer-oriented statement and billing purposes. On-Line Enquiry permits information for Customer Service and Credit departments. Interface to General Ledger provided if both systems used. Requires CBASIC .... **$699/$25**
- **Accounts Payable** — Provides aged statements of accounts by vendor with check writing for selected invoices. Can be used alone or with General Ledger and/or with NAD. Requires CBASIC .... **$699/$25**
- **NAD** Name and Address selection system — interactive mail list creation and maintenance program with output as full reports with reference data or restricted information for mail labels. Transfer system for extraction and transfer of selected records to create new files. Requires CBASIC .... **$79/$20**
- **QSORT** — Fast sort/merge program for files with fixed record length, variable field length information. Up to five ascending or descending keys. Full back-up of input files created .... **$95/$20**

GRAHAM-DORIAN SOFTWARE SYSTEMS

- **PAYROLL SYSTEM** — Maintains employee master file. Computes payroll withholding for FICA, Federal and State taxes. Prints payroll register, checks, quarterly reports and W-2 forms. Can generate ad hoc reports and employee form letters with mail labels. Requires CBASIC. Supplied in source code .... **$590/$35**
- **APARTMENT MANAGEMENT SYSTEM** — Financial management system for receipts and security deposits of apartment projects. Captures data on vacancies, revenues, etc. for annual trend analysis. Daily report shows late rents, vacancy notices, vacancies, income lost through vacancies, etc. Requires CBASIC. Supplied in source code .... **$590/$35**
- **INVENTORY SYSTEM** — Captures stock levels, costs, sources, sales, ages, turnover, markup, etc. Transaction information may be entered for reporting by salesman, type of sale, date of sale, etc. Reports available both for accounting and decision making. Requires CBASIC. Supplied in source code .... **$590/$35.**
- **CASH REGISTER** — Maintains files on daily sales. Files data by sales person and item. Tracks sales, overrings, refunds, payouts and total net deposits. Requires CBASIC. Supplied in source code .... **$590/$35**

MICRO FOCUS

- **CIS COBOL** — Version 3 is ANSI 74 subset with extensions which offer powerful interactive screen formatting and built in cursor control. Version 4 additionally offers full level 1 ANSI for Nucleus, Table Handling, Sequential Relative and Indexed I/O, Inter-Program Communication and Library
  .... Version 3, **$650/$50**
  .... Version 4, **$850/$50**
- **FORMS** — Interactive utility to create CIS COBOL source code to perform CRT screen handling in application programs. Supports full prompt text, protected fields and input validation against data type and range expected .... **$150/$15**
  When purchased with CIS COBOL .... **$125/$15**

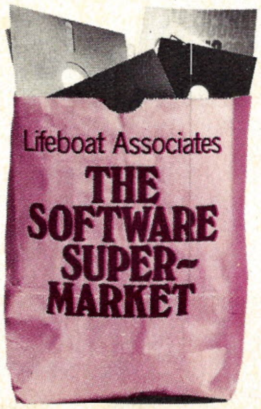
OTHER

- **tiny C** — Interactive interpretive system for teaching structured programming techniques. Manual includes full source listings .... **$75/$40**
- **C Compiler** — Supports most major features of language, including Structures, Arrays, Pointers, recursive function evaluation, linkable with library to 8080 binary output. Lacks data initialization, long & float type and static & register class specifiers. Documentation includes "C" Programming Language book by Kernighan & Ritchie .... **$110/$15**
- **ALGOL 60 Compiler** — Powerful block-structured language featuring economical run time dynamic allocation of memory. Very compact (24K total RAM) system implementing almost all Algol 60 report features plus many powerful extensions including string handling direct disk address I/O etc. Requires Z80 CPU .... **$199/$20**
- **Z80 Development Package** — Consists of: (1) disk file line editor, with global inter and intra-line facilities; (2) Z80 relocating assembler, Zilog/Mostek mnemonics, conditional assembly and cross reference table capabilities; (3) linking loader producing absolute Intel hex disk file .... **$95/$20**
- **Z80 Debugger** — Trace, break and examine registers with standard Zilog/Mostek mnemonic disassembly displays. Facilities similar to DDT. $35 when ordered with Z80 Development Package .... **$50/$10**
- **DISTEL** — Disk based disassembler to Intel 8080 or TDL/Xitan Z80 source code, listing and cross reference files. Intel or TDL/Xitan pseudo ops optional. Runs on 8080 .... **$65/$10**
- **DISILOG** — As DISINTEL to Zilog/Mostek mnemonic files. Runs on Z80 only .... **$65/$10**
- **TEXTWRITER II** — Text formatter to justify and paginate letters and other documents. Special features include insertion of text during execution from other disk files or console, permitting recipe documents to be created from linked fragments on other files. Ideal for contracts, manuals, etc. Also creates form letters in conjunction with a mail list file .... **$75/$10**
- **TEXTWRITER III** — All the features of TW II plus generation of table of contents, sorted index and handling of footnotes. TW II users may upgrade for $60. **$125/$10**
- **WHATSIT?***** — Interactive data-base system using associative tags to retrieve information by subject. Hashing and random access used for fast response. Requires CBASIC .... **$125/$25**
- **XYBASIC** Interactive Process Control BASIC — Full disk BASIC features plus unique commands to handle bytes, rotate and shift, and to test and set bits. Available in Integer, Extended and ROMable versions.
  Integer Disk or Integer ROMable .... **$295/$25**
  Extended Disk or Extended ROMable .... **$395/$25**
- **SMAL/80** Structured Macro Assembled Language — Package of powerful general purpose text macro processor and SMAL structured language compiler. SMAL is an assembler language with IF-THEN-ELSE, LOOP-REPEAT-WHILE, DO-END, BEGIN-END constructs .... **$75/$15**
- **Selector II** — Data Base Processor to create and maintain single Key data bases. Prints formatted, sorted reports with numerical summaries. Available for Microsoft and CBASIC (state which). Supplied in source code .... **$195/$20**
- **Selector III** — Multi (i.e., up to 24) Key version of Selector II. Comes with applications programs including Sales Activity, Inventory, Payables, Receivables, Check Register, Expenses, Appointments, and Client/Patient. Requires CBASIC. Supplied in source code .... **$295/$20**
  Enhanced version for CBASIC-2 .... **$345/$20**
- **CPM/374X Utility Package** — Has full range of functions to create or re-name an IBM 3741 volume, display directory information and edit the data set contents. Provides full file transfer facilities between 3741 volume data sets and CP/M files .... **$195/$10**
- **Flippy Disk Kit** — Template and instructions to modify single sided 5¼" diskettes for use of second side in single sided drives **$12.50**
- **BASIC Comparison** — A comprehensive features and performance analysis of five 8080 disk BASIC languages — CBASIC, BASIC-E, XYBASIC, Microsoft Disk Extended BASIC, and Xitan's Disk BASIC. Itemizes results of 21 different benchmark tests for speed and accuracy and lists instructions and features of each BASIC .... (send 20¢ S.A.S.E.) **FREE**

*CP/M is a trade name of Digital Research
**Z80 is a trademark of Zilog, Inc.
***WHATSIT? is a trademark of Computer Headware.

†CP/M for Heath, TRS-80 Model I and PolyMorphic 8813 are modified and must use especially compiled versions of system and applications software.
††PolyMorphic 8813 CP/M scheduled for September 15 release.

Software for most popular 8080/Z80 computer disk systems including **NORTH STAR, MICROPOLIS, iCOM, DYNABYTE DB8/2, SD SYSTEMS, ALTAIR, EXIDY SORCERER, VECTOR MZ, 8" IBM, HEATH H17 & H89, HELIOS, IMSAI VDP42 & 44, REX, POLYMORPHIC 8813†** *and* **OHIO SCIENTIFIC** *formats*



Orders must specify disk systems and formats: e.g. North Star single or double density, IBM single or 2D/256, Altair, Helios II, Micropolis Mod I or II, 5¼" soft sector (Micro iCOM/SD Systems Dynabyte), etc.

Add $1/item shipping ($2 min.). Add $1 additional for UPS C.O.D.

Manual cost applicable against price of subsequent software purchase.

The sale of each propriety software package conveys a license for use on one system only.

Shopping List No.5

EFFECTIVE JULY 15

Lifeboat Associates, 2248 Broadway, N.Y., N.Y. 10024
(212) 580-0082 Telex: 668585

™The Software Supermarket is a trademark of Lifeboat Associates

5-STAR EDITION for users of Radio Shack's TRS-80*

★ ★ ★ ★ ★

5-STAR EDITION for users of Radio Shack's TRS-80*

TRS-80 TRANSFORMED WITH PROFESSIONAL SOFTWARE PACKAGES

Lifeboat Associates, specialists in microcomputer disk software, is proud to offer the first professional disk-based language and utility package for the Radio Shack TRS-80 computer. Written by Microsoft, creators of Level II BASIC, the package runs on a TRS-80 system with 32K RAM, one or more drives and TRSDOS. The software is supplied on diskettes and consists of:

FORTRAN a true relocatable machine code compiler for ANSI FORTRAN X3.9 (except COMPLEX variables).

MACRO ASSEMBLER a disk-based macro assembler utilizing Zilog mnemonics and producing relocatable code.

LINKING LOADER to link-edit and load FORTRAN and assembler modules for execution.

SUBROUTINE LIBRARY a complete library of subroutines existing as relocatable linkable modules for FORTRAN or assembler programs—e.g., double precision square root, natural log, transcendentals, etc.

DISK TEXT EDITOR to create and modify FORTRAN and assembler programs as disk files: also can be used as a general purpose text editor for correspondence and other documents.

This high-powered professional software package with full documentation is available at the DISCOUNT PRICE OF **$150.** PER COMPUTER SYSTEM

The Macro Assembler, Loader, Editor, and Cross Reference Utilities alone . . **$80**

The Fortran Compiler, Loader, Editor, and extensive library of scientific functions alone . . **$80**

AND MUCH MORE

Software with Manual / Manual Alone

CP/M-operating system modified for use with TRS-80 computer and disks. In addition to the standard CP/M utilities of Editor, Assembler, Debugger etc., we have added: DCV2 (Utility to convert system tapes to CP/M files.) DISKAS & CASDISK (Utilities to back up files to tape and recover to disk.) MOVER (Program to transfer files with single drive systems) **$145/$25**

All items listed below operate in conjunction with the CPM operating system.

MAC — 8080 Macro Assembler. Full Intel macro definitions. Pseudo Ops include RPC, IRP, REPT, TITLE, PAGE, and MACLIB. Z-80 library included. Produces Intel absolute hex output plus symbols file for use by SID (see below) **$100/$15**

SID — 8080 symbolic debugger. Full trace, pass count and break-point program testing system with back-trace and histogram utilities. When used with MAC, provides full symbolic display of memory labels and equated values ...... **$85/$15**

TEX — Text formatter to create paginated, page-numbered and justified copy from source text files, directable to disk or printer .... **$85/$15**

DESPOOL — Program to permit simultaneous printing of data from disk while user executes another program from the console .... **$50/$1**

Disk Extended BASIC — Version 5, ANSI compatible with long variable names, WHILE/WEND, chaining, variable length file records .... **$300/$25**

BASIC Compiler — Language compatible with Version 5 Microsoft interpreter and 3-10 times faster execution. Produces standard Microsoft relocatable binary output. Includes Macro-80. Also linkable to FORTRAN-80 or COBOL-80 code modules .... **$350/$25**

FORTRAN-80 — ANSI '66 (except for COMPLEX) plus many extensions. Includes relocatable object complier, linking loader, library with manager. Also includes MACRO-80 (see below) .... **$400/$25**

COBOL-80 — ANSI '74 Relocatable object output. Format same as FORTRAN-80 and MACRO-80 modules. Complete ISAM, interactive ACCEPT/DISPLAY, COPY, EXTEND .... **$625/$25**

MACRO-80 — 8080/Z80 Macro Assembler. Intel and Zilog mnemonics supported. Relocatable linkable output. Loader, Library Manager and Cross Reference List utilities included .... **$149/$15**

EDIT-80 — Very fast random access text editor for text with or without line numbers. Global and intra-line commands supported. File compare utility included .... **$89/$15**

PAYROLL SYSTEM — Maintains employee master file. Computes payroll withholding for FICA, Federal and State taxes. Prints payroll register, checks, quarterly reports and W-2 forms. Can generate ad hoc reports and employee form letters with mail labels. Requires CBASIC. Supplied in source code. .... **$605/$35**

APARTMENT MANAGEMENT SYSTEM — Financial management system for receipts and security deposits of apartment projects. Captures data on vacancies, revenues, etc. for annual trend analysis. Daily report shows late rents, vacancy notices, vacancies, income lost through vacancies, etc. Requires CBASIC. Supplied in source code. . **$605/$35**

INVENTORY SYSTEM — Captures stock levels, costs, sources, sales, ages, turnover, markup, etc. Transaction information may be entered for reporting by salesman, type of sale, date of sale, etc. Reports available both for accounting and decision making. Requires CBASIC. Supplied in source code. .... **$605/$35**

CASH REGISTER — Maintains files on daily sales. Files data by sales person and item. Tracks sales, overrings, refunds, payouts and total net deposits. Requires CBASIC. Supplied in source code .... **$605/$35**

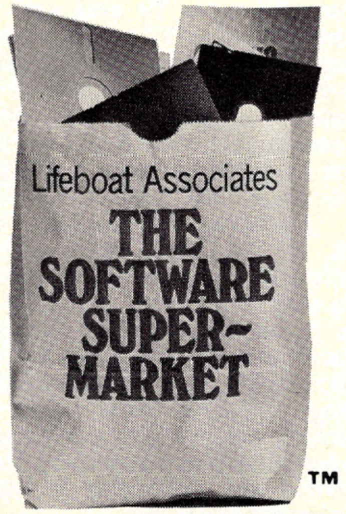
CBASIC-2 Disk Extended BASIC — Non-interactive BASIC with pseudo-code compiler and runtime interpreter. Supports full file control, chaining, integer and extended precision variables etc. .... **$90/$15**

Flippy Disk Kit — Template and instructions to modify single sided 5¼" diskettes for use of second side in singled sided drives **$9.75**

Selector III — Multi (i.e. up to 24) Key Data Base Processor. Comes with applications programs including Sales Activity, Inventory, Payables, Receivables, Check Register Expenses, Appointments, and Client/Patient. Requires CBASIC Supplied in source code. Enhanced version for CBASIC-2 . . **$345/$20**

G2 Level III BASIC by Microsoft

Powerful extensions to Level II BASIC including 10 machine language user calls, long error messages, keyboard debounce, graphics commands and much more. Price includes User Manual, a Quick-Reference Card and a pre programmed cassette tape. **$45**



2248 Broadway, New York, N.Y. 10024
Telex: 668-585
(212) 580-0082

Lifeboat Associates
2248 Broadway
New York, N.Y. 10024

Telex: 668-585

Please send the following:

Software		Price
	☐ manual alone	
	☐ manual alone	
☐ Check ☐ UPS C.O.D.	shipping	
☐ Visa ☐ MasterCharge	$1. for C.O.D.	
	Total	

Account # ________ Exp. Date ________

Signature ________

My computer configuration (specify disk system) ________

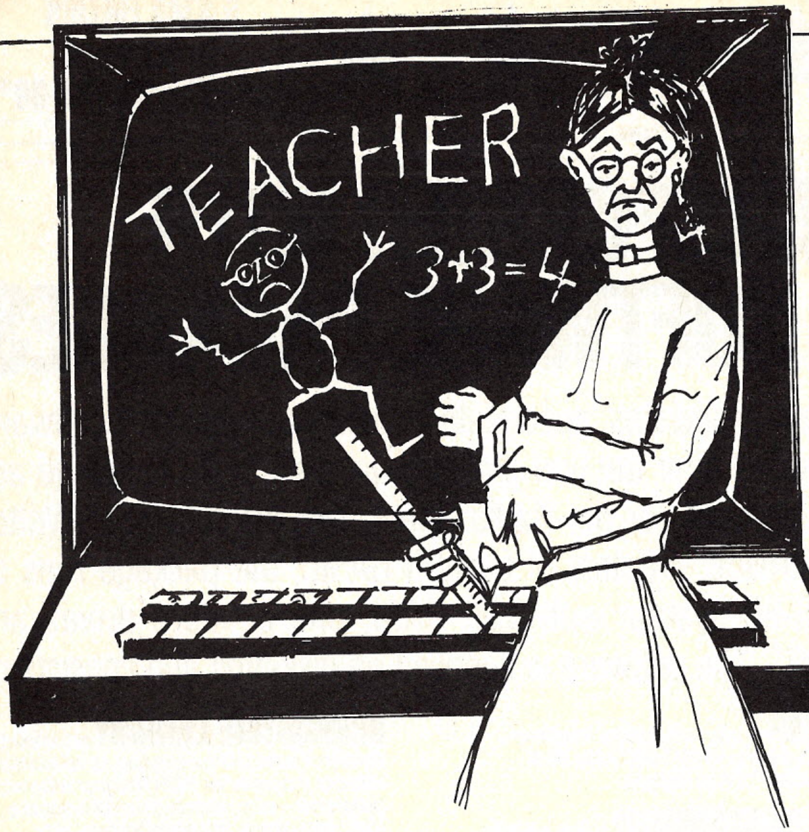
Name ________

Address (No P.O. Box) ________

City ________ State ________ Zip ________

CC

*Radio Shack and TRS-80 are trademarks of Tandy Corporation.



Classroom Computers: Beyond the 3 R's

Fred Bell

The classroom computer should, and can, go far beyond rote computer-aided instruction by teaching the student to analyze, evaluate and develop complex skills. Perhaps, as a result, the long-awaited "revolution in education" will be here sooner than predicted.

Since 1965. computer buffs, myself included, have been promising a revolution in education because computers are going to school. But where is this revolution? Certainly there has been at least a modest learning revolution; this is apparent from the many people who are learning about computers and using them to learn other things. In spite of the fact that some worthwhile applications are being done with computers in a few exemplary schools, this learning revolution has yet to take place in most schools. There has been an evolution in school learning (at least in many schools) that can be attributed, in part, to computer technology, but no real revolution.

Before considering the potential revolutionary effects of personal computers upon education, it is helpful to differentiate between school learning and out-of-school learning. The two are not always the same. We tend to learn things away from school when we want or need to learn them and we do so in our own way and at our own speed. This kind of learning has advantages and disadvantages. One advantage comes from higher motivation which encourages more inspired and efficient learning. On the other hand, the tendency to avoid difficult or uninteresting tasks may result in not learning some very useful and important things. Consequently schools are useful in coercing students, hopefully in a friendly and interesting way, into learning some things that are good for them which may not be learned otherwise. Out-of-school learning can be both good and bad, but so can in-school learning, which gets us to personal-computers and the education revolution.

Fred Bell, Professor of Mathematics Education, University of Pittsburgh, Division of Teacher Development, 4A01 Forbes Quadrangle, Pittsburgh, PA 15260.

Personal Computers and Dollars

One of the big reasons why personal computers may catalyze a revolution in our schools is that they are relatively cheap and should get even cheaper. Any family that can afford two color TV sets can now afford one color TV and a personal computer. Of course, any high school that could scrape up $10,000 per year for each of 10 years from a $1,000,000 per year budget could have had nearly all of its students using a minicomputer since 1969. (See James Saunder, **Mathematics Teacher**, May, 1978, pp. 443-447.) Fortunately, a family's decision-making processes in buying a personal computer are less cumbersome than a school's. Unfortunately for school students, as David Lichtman found (**Creative Computing**, January, 1979, page 48), educators are less enthusiastic about the computer's role in society and its potential for improving education than the general public.

But now, with low-cost personal computers, good computer applications may increase in schools. Home-computing enthusiasts have already begun to take learning out of the schools and are putting some of it back into the home where it belongs. Conversely, as more and more personal computers come to school, teachers can bring some of this good "street learning" back into the schools for the benefit of all students. Only $500 remaining in an equipment-and-supplies account at the end of the fiscal year can buy the first of many personal computers for student and teacher use.

History shows that many technological innovations that could be quite useful in promoting learning in schools do not get much use in schools until after they are common in homes and on the streets; for example, TV sets, audio recorders and hand-held calculators. Now that personal computers are "on the streets," we are beginning to see them filtering into schools. But will they be able to revolutionize education in schools? TV sets, audio recorders, calculators, and even minicomputers, while affecting what goes on in schools, failed to revolutionize education. Can we expect the personal computer to become a revolutionary agent? Yes, I think we can.

Personal Computers and Motivation

One of the most serious problems in schools is that of motivating stu-

Classroom, con't...

dents; that is, making them want to learn what teachers try to teach. The motivation problem occurs because sometimes teachers want to teach things (for good reasons) to students who do not particularly care to learn at the time. Perhaps the best hope for motivating students to learn in school is to pay attention to the nature of out-of-school learning. It appears that people learn non-survival things away from school for several reasons: (1) People learn to make things work (such as cars and computers) because they like to have control over impressive machines. (2) People learn to build model airplanes, radios, bookcases, etc. because they find satisfaction in creating something from nothing, or next to nothing. (3) People teach classes, give speeches, and write articles because they like to share their opinions and knowledge with others and possibly influence other people's opinions. Many things are learned because people enjoy the recognition and approval of other people. (5) Other activities that are not necessary for survival are carried out for relaxation, enjoyment, and self satisfaction.

But why do so many students dislike learning in school? First, students seldom have control over the academic machinery of schools; that is, the classroom learning environment. Second, creating and building tangible things occurs all to seldom in most classes. Third, students' opinions tend to be overshadowed by teachers' opinions in many classrooms. Fourth, many students get low grades in school, which interferes with their quest for recognition and approval. Fifth, much of what students have to do in school is neither relaxing, enjoyable, nor self-satisfying.

But how can a few personal computers in a classroom solve these motivational problems for students and teachers? Well, computers and computer-enhanced learning are not educational panaceas, but they can give students some real control over what they learn and how they learn it. Making a computer (an electronic monster) do one's bidding is fun for many people, in spite of the fact that it is, at times, tedious and frustrating. Writing a computer program and making it do what it is supposed to do is creating something — both a physical and an intellectual creation. Most people (including teachers and students) are impressed by good interactive computer games, simulations and tutorials, which provide recognition and influence for their creators. Finally, messing around, in a meaningful way of course, with a personal computer can be relaxing and enjoyable, in spite of many minor, temporary frustrations and aggravations.

Therefore, we find that personal computers in the hands of students in school can remove some of the artificial constraints of typical classroom environments and replace them with some of the personal freedoms inherent in many non-school learning situations.

Personal Computers and Learning

What is learned in school? English, reading, writing, arithmetic, French, history, etc.? Yes, these **are** some of the subjects that are taught but students should learn many other things that subsume all subjects. That is, students need to study each subject in a manner that permits them to function at all of the following cognitive levels:

- knowledge
- understanding
- application
- analysis
- synthesis
- evaluation
- problem solving
- knowing how to learn
- creating knowledge

Schools are fairly good at imparting knowledge (i.e., "George Washington was the first U.S. president") and understanding (i.e., "2 + 3 = 5 because 2 marbles together with 3 marbles is 5 marbles"). However, schools are only moderately successful at teaching applications (out-of-school uses for each subject), analysis (breaking a skill or conceptual structure into its parts), synthesis (building complex skills or conceptual structures from simpler things), and evaluation (comparing skills and structures and making judgments about them). Schools and teachers have even less success at teaching students the skills and heuristic procedures of problem solving, how to learn independently of teachers and courses, and ways of conducting the research and explorations that go into creating knowledge.

During the past 15 years we have demonstrated, through many dramatic examples, that computers can be used in schools to help teach knowledge, understanding, and applications of various subjects—things that were being done fairly well without computers. This is the evolutionary aspect of computers in education. But what about the higher-level cognitive activities, those things that we haven't been able to teach very successfully in school? Herein lies the true power of computers (especially personal computers) to really revolutionize learning and teaching in schools.

Writing a computer program requires analysis and synthesis of the subject under consideration as well as the program itself. A student cannot write a program to tutor others, play a game, simulate a situation, or solve an exercise without analyzing the topic being studied and synthesizing it into a coherent teaching/learning program. The synthesis required in writing the program properly and the analysis in debugging it provides additional practice at synthesizing and analyzing. Since many non-tutorial computer programs are higher-lever applications of topics, the student programmer must evaluate the approriateness of alternative approaches to the topic and the program. When a student writes computer programs to extend and clarify topics in school, the six steps in problem-solving (posing the problem, precisely defining the problem, gathering information, developing a solution strategy, finding the solution and checking the solution) must be carried out. On the other hand, most so-called "problems" in textbooks are really exercises for practicing skills, which require only one of the six steps of problem solving; namely, finding the answer. After several years of working with people in Project Solo at the University of Pittsburgh, we found that many students and teachers could carry out independent research of their own choosing in computer-enhanced learning environments. That is, these people were creating knowledge and learning how to learn independent of people who were labeled as the teachers and rooms that were called classrooms.

Now personal computers can bring the Solo concept of high-level, self-motivated learning out of the research-and-development laboratory

Classroom, con't...

and put it in the hands of large numbers of students and teachers in school classrooms.

Carrying Out the Revolution

Even before the advent of personal computers (as early as 1972), the computer technology and courseware existed for a revolution in teaching and learning in schools. Now personal computers with their low costs, easy accessibility, total dedication to the user, and person-on-the-street popularity may provide the long-awaited catalyst that is needed to make some dramatic changes in how computers are used in schools. In a few years large numbers of students entering high school will be as familiar with a computer as they are now with a TV set, probably more so since they will have actively programmed a computer, in comparison to watching television passively.

As a consequence of the popularity of television, Americans are accused of having become spectators rather than participants in life. Personal computing certainly requires active intellectual participation on the part of the user. I have yet to hear of anyone dozing off while sitting in front of a personal computer.

For several years mathematics teachers worried about whether kids should be allowed to use hand-held calculators in school. The popularity of calculators outside school quickly settled that issue. Nearly every family had a calculator. Pre-school children played with them and students brought them to school. Teachers could not ignore calculators because it was impossible to keep them out of school; so now they are trying to determine how best to incorporate calculators and calculator-related skills into the school mathematics curriculum. Even if people try to keep personal computers out of schools, they are going to fail. In a few years, when they are more efficiently packaged and even less expensive, personal computers can fill the "lunch-box-technology" void created by school-lunch programs. Instead of a lunchbox, students will be carrying a PET or TRS-80 computer on a handle to school. When this time comes, an Apple for the teacher will really help a kid get a better grade in school. □

References

Barstow, Daniel. "Computers and Education: Some Questions of Values," **Creative Computing** 5: 116-119; February 1979.

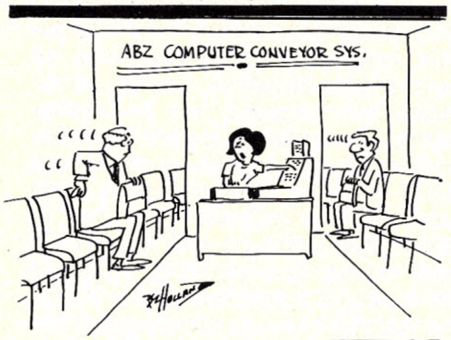
Bell, Frederick H. "Why Is Computer-Related Learning So Successful", **Educational Technology** 14:15-18; December 1974.

Bell, Frederick H. "Can Computers Really Improve School Mathematics?", **The Mathematics Teacher** 71: 428-33; May 1978.

Dwyer, Thomas A. "Soloworks: Computer-Based Laboratories for High School Mathematics," **School Science and Mathematics** 75:93; January 1975.

Lichtman, David. "Survey of Educator's Attitudes Toward Computers," **Creative Computing** 5:48; January 1979.

Molnar, Andrew R. "The Next Great Crisis in American Education: Computer Literacy," **THE Journal** 5:35-38; July/August 1978.

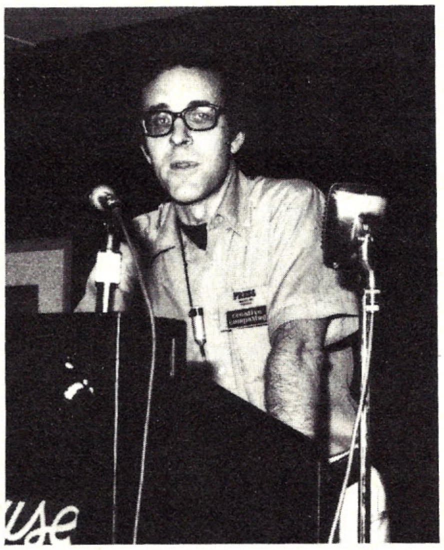


"Mr. Axilbroad will see you now, sir. Keep your seat."

Scenes from the Clemson University Conference on Small Computers

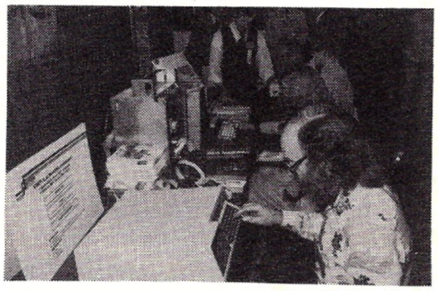
Dean Lyle Wilcox of the College of Engineering opened the conference and welcomed the over 600 attendees to Clemson.

Sessions were well attended, a tribute to Bill Barnett's and J.K. Johnson's excellent organization and publicizing of the conference.



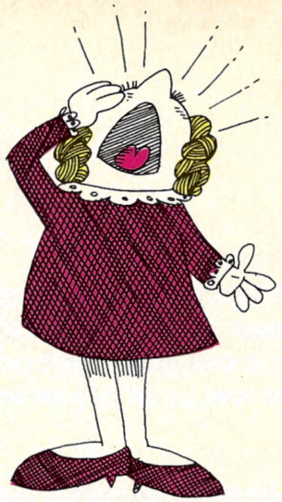
Keynote speaker David Ahl spoke about "microcomputer applications: the myth and the reality." Key conclusion: although technology is making more things possible and lowering prices, right now is the time to get involved with a micro.

Creative's display table featured magazines (of course) and new Apple II software releases.



A substantial number of attendees were businesspeople, engineers and professionals from throughout South Carolina and Georgia.

That queasy feeling in the stomach, the sweaty palms, the blank stare or preferring to watch over someone else's shoulder are all feelings familiar to the first-time computer user.



Computer Anxiety: One Way To Handle It

Anne H. Knight

We hear so much about math anxiety and nothing about computer anxiety...yet are we not faced with Orwell's "1984" and Toffler's "Future Shock?" Should not the real issue be one of handling technological anxiety? Knowing the frustration and anxiety of trying to keep up with a rapidly changing, highly technological society, we can argue that the technologically anxious person is primarily suffering from future shock.

Mathematicians have captured the market and the public interest with the concepts of math anxiety and math avoidance. Educators are beginning to recognize the existance of such a phenomenon. But who is raising the question of computer anxiety? Are the computer scientists and public school teachers even concerned? How are teachers preparing the young for the present shock of modern technology? Educational institutions appear to be delinquent in preparing themselves for dealing with this technological society. So we ask if the schools are failing us and our children by not introducing computers in the early learning stages. Many of my friends, who are over 30, recognize their anxiousness about computers. They know they have uncomfortable feelings when I talk about my work with computers. To meet the challenge of preparing the younger generation for the world around them, educators must overcome their own anxieties and spend time learning about computers and their capabilities.

Although the slogan "computer anxiety" may not become the buzzword for persons avoiding the use of computers, the concept is easily defined. That queasy feeling in the stomach, the sweaty palms, the blank stare or preferring to watch over someone else's shoulder are all feelings familiar to the first-time computer user.

Anne H. Knight, Coordinator of Educational Services, Computer Services, University of New Hampshire, Durham, N.H. 03824.

Borrowing liberally from the title of Shiela Tobias' book, **Overcoming Math Anxiety**, I selected "Overcoming Computer Anxiety Through Games and Simulations" as the title for my work session at the 57th Annual Convention of the National Council of Teachers of Mathematics in Boston, last April. My only instructions were to give a lot of opportunity for hands-on experience during the 90 minute work session.

As implied by the title of my work session I believe that the playing of games interactively at a computer terminal reduces a person's anxiety and increases his/her comfort and ease with the machine. The session I ran actually went beyond game playing and explored several instructional techniques.

First, I assumed that most of the participants would be unfamiliar with computers and inexperienced in using them. If they were not novices themselves, I could assume they would be instructing novices. Using four different techniques I was able to explain or demonstrate what a computer is, what it can do and how to do it. Finally, the attendees had a chance to try it. Before leaving the room, each participant was asked to respond to a questionnaire in order to evaluate which technique was most effective for him/her. Their responses endorsed my hypothesis that hands-on game playing was the most effective method for dispelling their anxiety.

By definition a NCTM work session meant little time for instruction and a lot of time for hands-on experience. There were 11 Radio Shack TRS-80's and 3 Digital Equipment Corporation ES210/C minicomputers with 4 terminals and a floppy disk drive attached to each. The cassette tapes selected were HURKLE, BAGELS, HMRABI, PLANTS, MALAR, DIETS, STOCK, BALPAY, ENERGY and LIMITS because of their general educational value and ease of operation. For the ES210/C there were 3 floppy disks which contained the 101 BASIC Computer Games adapted to the mini. Signs posted beside the terminals and printouts of sample runs limited the selection of programs to those mentioned.

As persons entered the room for the 90 minute session, they were directed to a micro or a terminal as space permitted. Some sat in pairs. Everyone seemed to have itchy fingers and curious looks showed on faces. However, I staved off their eagerness by outlining my presentation and telling them the hands-on time would be forthcoming. My intent was to provide an opportunity for persons to get in touch with their anxiety. I would be offering a carefully sequenced set of techniques to accomplish this. First would be the lecture method - familiar to all. Second would be a simple programmed learning sequence. Third would be the examination of a demonstration run of a program; and last, would be the opportunity to play games on the computer.

The lecture discussed the computer age, the sources of computer anxiety in adults and the instructional value of games. In order to remove the barrier to understanding the nature of a computer, a diagram was used to show graphically the separate components of a computer system.

The programmed text was a COPY-ME section of Calculators/Computers Magazine (DYMAX October 1977) called "Bits and Bytes" by Bob Albrecht which explained the internal language of a computer - i.e., binary digits and words. The exercise served to clarify the meaning of a memory location. The computer language of codes, acronyms and symbols was also mentioned as a barrier to understanding and cause of anxiety.

Anxiety, con't...

The sample run handed out was of BALPAY, a balance of payments simulation. This decision-making game was chosen because it is cross disciplinary, interesting and easy to grasp. In order to help them get in touch with their feelings or anxieties and to analyze the sample run, I used a paired discussion technique. After first examining the sample run alone, the participants talked in pairs about what the printout meant. Then they formed small groups of 4 or 6 to further clarify the program's meaning and discuss how they felt about reading the printout. Finally, one person from each group reported in turn to the whole group what was discussed within their small groups as to feelings about the printout.

The first three methods took 50 minutes. Then during the last 40 minutes each person at a micro or terminal could run the available computer game. People could switch to different programs by moving to different machines. Assistance was available in the peculiarities of the different machines and/or programs even though the difference between the machines was obscure to most of the users. The games were all easy enough to run yet challenging enough to create interest.

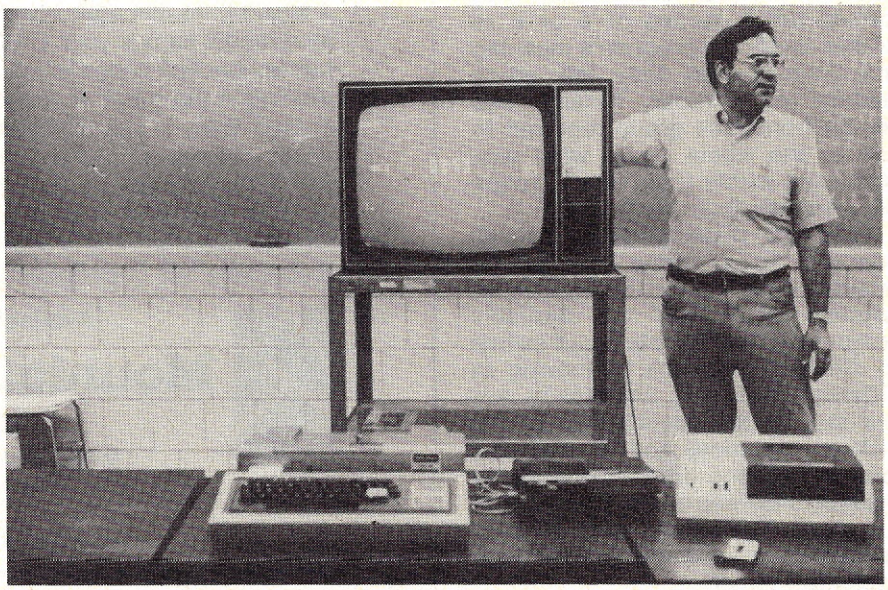
Before leaving the session each person responded to the questionnaire I distributed. Many persons responded with a thank you and one woman responded that she had "no anxiety left."

It may not have been the hands-on game-playing alone that dispelled the anxiety this woman felt. It may have been the combination of all four steps I used during the work session. But the fact remains that many people experience fear and uncertainty about using a computer. There are a variety of techniques for addressing computer anxiety. Some are machine-based - such as the game. Games usually proliferate on a computer since game programs are part of the initial software package acquired with a new computer. Certainly their use should be encouraged since they are most effective in easing the tension one experiences when approaching a computer for the first time. However, I feel these machine-based activities should be combined with non-machine-based activities, such as paired sharing and group discussions. This approach to dealing with computer anxiety should ease the transition into today's highly technological society for persons avoiding the use of a computer. □

The TRS-80 As A Classroom Teaching Tool

Dr. Samuel W. Spero

The computer can become a tremendous teaching aid in classrooms ... if teachers will just let it.



As a result of considerable investigation and research we have discovered a computer-based classroom teaching tool which is not only cost-effective but also has tremendous potential for improving classroom instruction. The computer system is based on the Radio Shack TRS-80 Level II System.

The System

The hardware configuration is as follows:

1	4K - Level II TRS-80	$499.00
1	Cassette Recorder	49.95
1	Expansion Interface	299.00
1	P-1 Printer	599.00

Note: We have purchased the Centronics P-1 Printer on special arrangement, however we have been informed that Radio Shack will soon be supporting the P-1 as a standard option for the TRS-80. The above numbers therefore may change.

1 Cable to connect TRS-80 to TV Monitor 20.00

Note: All of the above can be purchased from any Radio Shack Dealer and also serviced there.

1 Olson Electronics Television Monitor 24" — #TV-352 199.00

Total Cost $1,665.95

If the school has its own monitor and cassette recorder, this reduces the cost of the system. Furthermore, if a school or school system owns a share of Tandy stock, it is entitled to a 10% discount on the above hardware.

An advantage of the TRS-80 versus other systems lies not in the hardware itself but rather in the fact that a school can get service for its TRS-80 through any Radio Shack dealer in the country. The importance of this point cannot be minimized. Beyond the 90-day warranty on the hardware, repairs will cost about $25.00. In over nine months, we have had no trouble with our machine. The above is a starter system which has plenty of versatility and can also be expanded. Through the expander box, a bigger printer can be attached along with more memory, a floppy disk, and many other peripherals can be added.

As Photo 1 shows, the system is highly portable and moves easily from classroom to classroom. If you also purchase the small TV monitor manufactured by Radio Shack, the system is portable enough to be moved from building to building and even from school to home. This latter is a real advantage for teachers who are willing to work at home evenings and weekends on developing software. This is a major advantage over time-sharing systems which do not exhibit such portability.

Applications

Before listing some of the applications which were developed by teachers with whom I worked this summer, a few words are in order as to the approach used. The total thrust of our activities has been and is to use the TRS-80 as an aid to, rather than as a replacement for, the classroom teacher. For example, the teacher uses the TRS-80 in class with the television monitor and the 32-characters-per-line option to help students understand problem-solving algorithms in junior and senior high school mathematics courses. The teacher uses it in the social studies classroom with the monitor to create a gaming framework in the study of the operation of our government (the Huntington II simulation game, POLICY). On the other hand it can be used by the teacher to generate problem sets (the printer operates at 180 characters per second) for individual students in mathematics and science or to generate themes for a composition or a word maze for vocabulary study in the English classroom. While Computer Assisted Instruction, where the student is tutored at the terminal by the

Dr. Samuel W. Spero, Dept. of Math, Cuyahoga Community College, 2900 Community College Dr., Cleveland, OH 44115.
Phone: (216) 241-5966.

computer, has not been discouraged, it has not been encouraged in work with our teachers. The reason for this attitude is that CAI is difficult to develop, dedicates the facility to a single student, and prevents use of the facility for other more cost-effective and pedagogically effective applications.

As with any new instructional tool, teacher training is crucial. We were very fortunate to obtain a grant under Gifted and Talented Program of the U.S. Office of Education. This grant provided us with the funding needed to train 20 junior high school teachers in the use of computers as a tool of instruction for gifted students. Simultaneously, we received a Title IV-B grant which was sufficient to purchase the TRS-80 system for each of the five schools represented at the workshop. Attending the workshop were teachers of mathematics, science, social studies, and English. The three week workshop was designed to help each of the teachers develop at least one computer-based activity or unit which would be incorporated into the existing curriculum or instructional strategies. Rather than "reinventing the wheel" I brought to class a collection of materials that had already been developed and suggested that teachers use these as the starting point for their units. Included were the Huntington I and II materials published by Digital Equipment Corporation, as well as back issues of CREATIVE COMPUTING, PEOPLE'S COMPUTERS and CALCULATORS/COMPUTERS MAGAZINE. The following projects were the result of the workshop.

The topics developed were:

Mathematics

A Classroom-Exercise for Factoring Trinomials.
Problem-generator for Decimal Computation.
Computer and the Study of Mathematics for 7th Grade.
Computer Literacy in the Math Classroom for 8th Graders.
Computer and Mental Arithmetic Exercises.
Statistics and the Computer in 7th Grade Math.
Elementary Graphics with the Computer.

Science

The Scientific Method - Computer Style.
Date Reduction in the Junior High School Science Lab.
Computer Help in Calculating a Water Budget (Adaption of Huntington I Application).
The Physiology of Cells - a Computer Based Experience (Adaption of Huntington I Application).

Social Studies

Computerized Madlibs and the Oregon Trail.
A Computerized Tutorial on the Passage of a Bill.
POLICY - A Simulation Game for Social Studies (Adaption of Huntington II Application).
A Computerized Gradebook.

English

WORDFIND Puzzles and English Vocabulary Study.
Producing HAIKU Poetry with the Computer - a Motivational Application.
Computerized Telephone Stories.
COMPUTER Education for Gifted and Talented Youths - A Computer Literacy Unit for an English Class.
The Computer: Fact and Fiction - A Literature Unit.
Use of the Computer for Out-of-class Individualized Quizzing.

These and other applications will be tested in the classroom this coming school year and the results published. Inquiries are welcome. □

The Computer vs. The Professor

Donald E. Mowrer, Ph.D.

Some observations on computer-assigned instruction and a CAI language called CHIMP.

A pocket calculator was my closest encounter with anything resembling a computer when I walked into the office of Max Ivy, Director of Computer Services for Arizona State University.

"I'd like to know if I could use the computer to deliver quizzes to students in my speech pathology class," I began.

"What languages do you know," he inquired?

"Well, I know a little French and German, but what's that got to do with it?"

In an instant, Max knew I was either a wise-guy or terribly naive about computer programming. Unfortunately, the second alternative proved to be the case.

That was four years ago. If you're interested in learning how the novice college professor gets initiated into computer programming, read on.

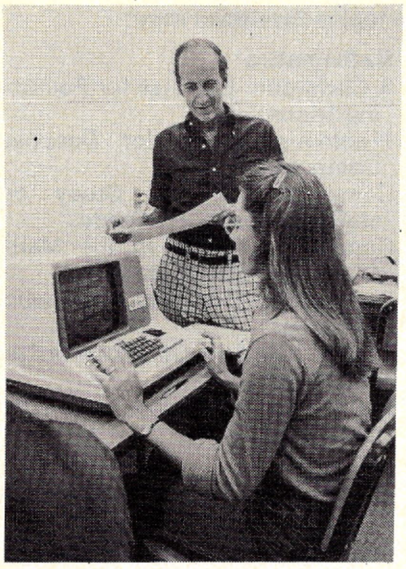
In an instant, Max knew I was either a wise-guy or terribly naive about computer programming. Unfortunately, the second alternative proved to be the case.

After considerable effort, and by considerable, I mean horrendous, I was able to put my quizzes into computer language. The computer started working for me for a change. But, it wasn't that way in the beginning.

Selecting A Language

Max did a lot of head scratching as he considered my request. How could he help an ambitious professor whose knowledge of computers was so limited? He recalled reading about a computer program, CHIMP, developed in 1969 by R. J. Munn and K. Wolff of the Institute for Molecular Physics and Science Teaching Center at the University of Maryland. It was designed for use on the Univac 1108 as an author language for novices like myself. That afternoon Max mailed a blank tape to the University of Maryland for a dub of the CHIMP program.

Donald E. Mowrer, Ph.D., Communication Dept., Arizona State University, Tempe, AZ 85281.



Author with student at CHIMP terminal.

While waiting for the tape, Max suggested I contact the few other faculty members who were involved in computer assisted instruction. I say a few because there were only four individuals who showed much interest in this area. He also suggested I secure a computer ID and account number so I could work my way through a BASIC language tutorial program on our Univac computer.

My first quest led me to the mathematics department where I discovered PLANIT, a complex author language that was beyond my understanding. It was also expensive to run, could be used with only one student at a time, and was not thoroughly debugged. Scratch PLANIT.

Another language, PLATO, was discussed as a possibility but it was as complex as PLANIT and was not available on our computer. Scratch PLATO.

A trip to the engineering department introduced me to APL. One of the professors used APL with his advanced students only and suggested that first I become thoroughly familiar with BASIC before attempting APL. Scratch APL.

The CHIMP system is designed to permit fairly natural communication between the pupil and the teacher.

By this time I was assigned an account number enabling me to run through the BASIC tutorital program at a terminal. I soon learned that BASIC was designed for those who wished to solve mathematical problems. When I finished explanations of IF THEN and GO TO statements, the suceeding instructions left me in a daze. What I really needed was a three hour semester course in BASIC before I could hope to write with that language. Scratch BASIC.

By the time the CHIMP program had arrived, I was convinced that computer access was reserved for engineers and mathematicians and nothing short of several courses could help me.

When CHIMP arrived, this feeling was quickly reversed. Line 54 of CHIMP's introduction stated, "It has been designed so that a faculty member who has little or no programming experience can write an interactive lesson with minimum effort." Whoopee! Finally, someone had taken pitty on the novice faculty member and had written instructions I could use.

CHIMP Basics

The CHIMP System is designed to: 1. interpret the author's lesson, 2. present material to the pupils, 3. receive, analyze, and act upon the student's replies in accordance with the author's goals. By following the simple directions in the CHIMP manual, an author can easily develop a lesson. The CHIMP system is designed to permit fairly natural communication between the pupil and the

teacher. The only special skill required of the student is the ability to sign on the computer with proper ID codes. All answers are typed in normal conversational speech script.

A lesson consists of a number of lines of textual material. Each line is divided into three parts. The first part of the line, columns 1 and 2, serve as an optional line label. It is often necessary to refer to specific lines (hence, the label) but many lines require no label. The second part consists of columns 5, 6 and 7, the operation code. This code specifies what the computer is to do with the line. The last part of the line occupies columns 10 through 72. It may consist of the author's text, signal to the student to provide an answer, or it may be left blank.

An author's lesson might look like this:

A1 PRE Who discovered America?
ANS

The computer would present the following lines to the student:

Who discovered America?
??

The student could type Columbus, Christopher Columbus, Columbus discovered America, or similar answer. Obviously, this PRE/ANS system limits the author's ability to provide feedback to the student's answer. Therefore, five operation codes are provided to perform different kinds of evaluations. The most common code is KEY (keyword). This code is followed by several other codes which specify certain features about the student's answer. For example, the first code, separated by a parenthesis, specifies if spelling mistakes are allowed. If the code is 1, then one letter can be altered. Columbus, Columbua, Cilumbus or Kolumbus would all be acceptable. A code of 2 would allow two spelling errors, 3 three errors and so forth. The second code after KEY specifies whether the key words must be in a particular order (Y = yes, N = no). If so, then Columbus Christopher as an answer to the lesson above would be scored as wrong. If order is not specified, any word order would be scored as correct, as long as the key words were present. The third code represents the number of key words the author selected. In our example above, two key words are possible, Christopher and Columbus. The fourth code specifies the number of these key words that must be in the student's answer. If the code is 1, either Christopher or Columbus will be accepted as a correct answer. If the code is 2, both words are required in the student's response. The remaining codes contain the key words associated with the answer.

If order is not specified, any word order would be scored as correct, as long as the key words were present. The third code represents the number of key words the author selected.

An expansion of the first lesson might look like this:

A/PRE Who discovered America"
ANS
KEY(1), Y, 2, 2, Christopher, Columbus

The KEY code tells us the author made the following decisions: the student can make one spelling error, Christopher must be followed by Columbus, two key words are available as answers, and two key words must be present.

Other KEY codes can be designed to accept specific answers or numerical answers that fall between specified limits.

Following the KEY code line, the author includes feedback for the student's correct answer, which might be, "That's right." A response to an incorrect answer is also included on another line.

```
A1    PRE    GREETINGS AGAIN, $NAME$.
             NOW THAT YOU KNOW EVERYTHING THERE IS TO
             KNOW ABOUT INSTRUCTIONAL PROGRAMS, I TAKE IT YOU ARE
             READY TO PROCEED.  IT WON'T BE AS BAD AS YOU THINK SINCE
             I'VE TESTED THE QUESTIONS OUT ON A LOT OF STUDENTS.
             THEREFORE, I SHOULD BE ABLE TO DETERMINE YOUR MASTERY
             OF THE SUBJECT.  I PRESUME YOU'RE READY TO GO
             SO LET'S GET STARTED.  ARE YOU READY?
      ANS
      KEY    N, 5, 1, YES, COURSE, CERTAINLY, SURE, NATURALLY
             FINE, LET'S PROCEED WITH FOUR FILL-INS.
      GTO*   S1-1
      GTC    C0, 0, *+1, W1
             UNLESS YOU TYPED IN SOME CUTE WAY OF SAYING YES,
             I WOULD ASSUME YOU AREN'T READY.  IF YOU ARE, PLEASE
             TYPE IN YES.
      CTR    C0=C0+1
      GTO    A1+7
W1    PRE    WELL, I TAKE IT YOU AREN'T READY THEN.  IF THAT'S TRUE
             I'LL SIGN OFF AND GIVE YOU TIME FOR MORE STUDY.
      END
      CTR    C0=0
S1    PRE    A CLINICIAN IS ASKING A CHILD, EARLY IN ARTICULATION
             DRILLS, TO REPEAT WORDS CONTAINING THE TARGET SOUNDS.
             SHE MAY SWITCH TO PRESENTATION OF PICTURES AND ELEMINATE
             THE REPETITION OF WORD DRILLS.  WHY IS THIS DONE?
      ANS
      KEY    N, 2, 2, REDUCE=FADE=ELEMINATE, CUE=STIMULUS=STIMULI=PROMPT
             THAT'S GREAT, $NAME$
      CTR    P0=P0+1
      GTO*   S2-1
      GTC    C0, 0, *+1, W2
             WANT TO TRY AGAIN?  YOU DIDN'T SAY IT QUITE CORRECTLY.
      CTR    C0=C0+1
      GTO    S1
W2    PRE    MISSED AGAIN, $NAME$, IT WAS TO REDUCE THE FADE OF CUES.
      CTR    C0=0
```

Figure 1. Sample CHIMP program.

Thus a simple lesson would look like this:

```
A4 PRE Phoenix is in what state?
ANS
KEY(1), N, 1, 1, Arizona.
 That's correct
*
UNX 1
 Sorry, it's in Arizona
*
A5 PRE What's the capitol of
Nevada?
```

CHIMP processes this lesson in the following manner. Line A4 is presented to the students and waits for an answer. Once the answer is executed, it is compared with the KEY. If the answer meets the KEY requirements, CHIMP is in a test-satisfied state and prints the line following KEY (That's correct) and skips to line A5 to present the next lesson. If CHIMP is in a test-failed state after processing the answer, it prints the line after UNX 1 (Sorry, it's in Arizona) and proceeds to line A5. If UNX is set at 2, line A4 will be presented again to give the student a second chance.

A simple dialogue of question answer formats can be carried out using this simple lesson. Using a GTO (Go to) label in place of the UNX, the student can be shifted to any line in the text. More information can be presented and the student can be given additional chances to answer the question. By using a counter, points can be added or subtracted from a student's score depending upon how many hints are provided or how many tries are allowed. The student can even opt to transfer into BASIC to perform mathematical computations.

A wide variety of other options including the use of macro units are also available in the program but they will not be discussed here. The point to be made is that the average teacher can develop effective lessons for students using the computer as the delivery vehicle. Little or no computer experience is required.

Getting the Programs Up

The procedure for writing and entering the lessons in CHIMP appears deceptively simple. The belief in its simplicity was my first mistake. I have been plagued with two problems from the onset. The first problem consisted of small programming mistakes I made while writing the lessons. I failed to include some codes where they were supposed to be, I forgot to add confirmation of the student's response, or some such stupid mistake. The second problem was mistakes made by the typist who entered the program on the computer. It was her first attempt at typing on a computer keyboard. She made several spacing and typing errors.

Debugging each of the ten quizzes before they could be administered to the students required more time than it took to write them. I'm not used to letter-perfect work but I soon learned that careful proofing is the key to making programs run correctly.

Nothing is more frustrating to a student than reading the computer's output, "sorry, you're wrong" when they know they typed in the correct answer.

If you only learn one thing from this article, I hope it will be the realization that the computer will tolerate no errors in the program. Those who are old hats at program writing know this but novice professors like myself do not.

Student Reaction

Once the program was up and running, I was ready to turn the students loose at computer terminals. First you should know that female students outnumber male students in speech pathology 20 to 1. At least 95% of them never sat in front of a computer terminal before and a fourth of them could not type. To say that they felt intimidated in the presence of a computer terminal is putting it mildly. The least problem caused absolute panic. Many simply left the terminal without signing off. Computer consultants unfamiliar with CHIMP could offer little help. One chief drawback was that only one student could execute the program at a time. Even though the students knew this they felt they were doing something wrong when they couldn't get on the computer. They pestered the consultant, cursed the computer screen, and complained to me when the computer wouldn't let them sign in the program.

A second major problem was that I did not anticipate some of the answers students would offer. Nothing is more frustrating to a student than reading the computer's output, "sorry, you're wrong" when they know they typed in the correct answer. These students came to my office infuriated because they lost points on an item they knew to be correct. Although I tried to make corrections in the program to accommodate variations of correct answers, I was never able to eliminate this problem.

While the immediate feedback provided by the computer is heralded as one of the chief advantages of computer assisted instruction (CAI), this feature may act as a disadvantage. The student who misses a question becomes apprehensive. The chances of getting an A begin to fade. A few more misses adds to anxiety feelings and soon they develop hostile feelings toward the computer. These are things you seldom read about in descriptions of CAI. Proponents of CAI are quick to point out the advantages such as savings of time, increased accuracy, and how it can meet individual needs. While these factors may hold true for the good student, I found my computer quizzes were a disaster for marginal and poor students in my classes.

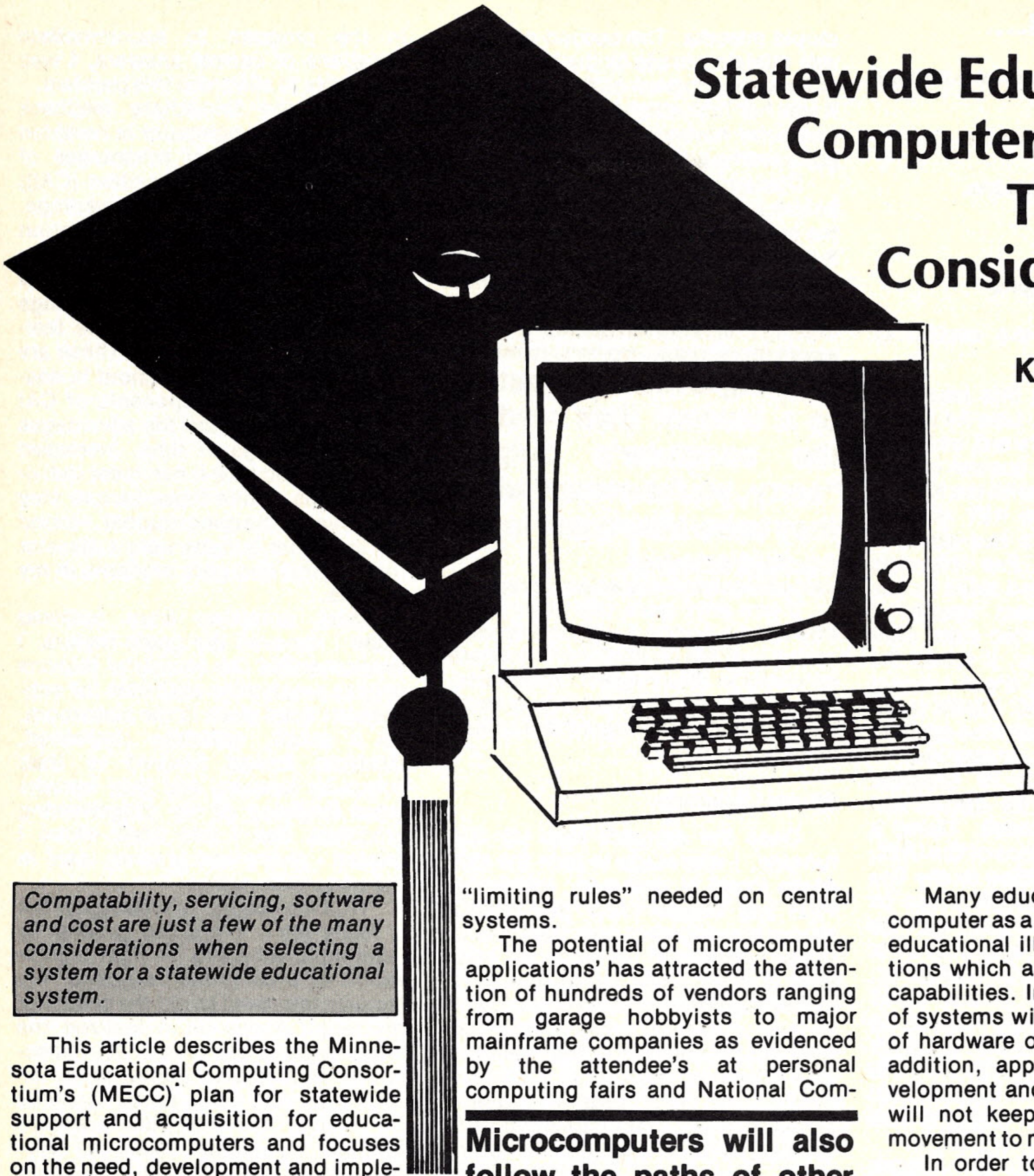
Most problems were resolved when I removed the grade feature. I recommended the computer lessons only as preparation activities for subsequent tests given in the classroom. As a result, anxieties decreased, students looked forward to their computer lessons, and they became more forgiving of my programming mistakes. But I was back where I started - giving and grading tests in the classroom again!

This may sound discouraging but it really isn't. I profited greatly by learning to adhere to the tight organization of a computer language. My lectures improved as did the quality of my tests. Above all, I realized the importance of my role in the classroom. That role is portrayed by an individual who cares about students in a way the computer cannot. Despite the claims made by proponents of programmed instruction who maintain students can learn just as well without instructors, results of research studies fail to support the supposed advantages of instructorless classes.

A combination of CAI and teacher instruction seems to result in an ideal instructional delivery system. Each can do certain things better than the other. What is most needed now is additional author languages that permit teachers to write instructional lessons on a wide variety of systems. CHIMP, for example, only works on Univac. Until these languages are available most teachers will abort attempts to use the computer.

Finally, college professors who develop computer delivered instructional materials can look forward to an exiting adventure. But take time to avoid the mistakes I made. □

Statewide Educational Computer Systems The Many Considerations



Kevin Hausmann

Compatability, servicing, software and cost are just a few of the many considerations when selecting a system for a statewide educational system.

This article describes the Minnesota Educational Computing Consortium's (MECC)* plan for statewide support and acquisition for educational microcomputers and focuses on the need, development and implementation of a plan.

Although the utilization of microcomputers in education is relatively new, within the next few years we can expect an exponential increase in the number of microcomputers sold to educational institutions. The growth is being spawned by a number of factors including the decreasing cost of microcomputers coupled with their increasing capabilities and the rapid growth of their use in a variety of fields throughout the country. Another important factor is the microcomputers' independence from a mainframe system which increases its portability and eliminates many communications-related problems, as well as the elimination of the "limiting rules" needed on central systems.

The potential of microcomputer applications' has attracted the attention of hundreds of vendors ranging from garage hobbyists to major mainframe companies as evidenced by the attendee's at personal computing fairs and National Computing Conferences. The resulting number of different systems makes it very difficult to stay abreast of current developments. Microcomputers will also follow the paths of other new technologies, meaning, many of the current microcomputer manufacturers may go out of business within a year or so, making it very important for microcomputer purchasers to be aware of manufacturer and vendor stability.

Microcomputers will also follow the paths of other new technologies, meaning, many of the current microcomputer manufacturers may go out of business within a year or so.

Many educators view the microcomputer as a panacea for a variety of educational ills resulting in expectations which are greater than system capabilities. In many cases, the use of systems will be impeded by a lack of hardware or software features. In addition, applications software development and instructional support will not keep pace with the initial movement to microcomputer usage.

In order to meet the needs and address the problems defined above, MECC set up a special task force to accomplish the following:

1. To conduct a survey for assessing the current and future microcomputer uses and needs of MECC users.
2. To determine the strengths and weaknesses of microcomputer utilization in various instructional computing modes and environments.
3. To provide demonstrations of microcomputer use for instructional purposes.
4. To coordinate and disseminate information regarding pilot programs using microcomputers.
5. To prepare recommendations regarding the potential for large scale acquisition and

Kevin Hausmann, Instructional Coordinator, Minnesota Educational Computing Consortium, 2520 Broadway Dr., St. Paul, MN 55113.

Statewide, con't...

utilization of microcomputers and the appropriate roles and responsibilities for MECC.

To begin addressing these objectives, the task force divided the tasks into three components: hardware, systems software, and applications software. Several Minnesota vendors were contacted and asked to supply systems for examination, evaluation and experimentation. Fourteen systems were evaluated by the task force.

Microcomputer Systems Evaluated

ALTAIR ATTACHE
ALTAIR 88-1301
ALTAIR MULTI-USER
ASTRAL 2000
IMSAI VDF - 80/1000
NCR 7200
OLIVETTI P6060
PET 2001
POLYMORPAICS 8813
PROCESSOR-TECH SOL
RADIO SHACK TRS-80
APPLE II (Integer BASIC)
TEKTRONIX 4051
TERAK RT-11

As a starting point, the task force defined a "minimal educational system." The minimal educational system must have:

a) A microprocessor.
b) I/O device(s). The system must include an ASCII keyboard and printer and/or monitor.
c) A permanent file storage device. This can be of the form: floppy disk, hard disk, etc.
d) A vendor supplied operating system.
e) The BASIC programming language must be supported.
f) At least 12K of user memory must be available. This excludes memory space required for the operating system and the language processor(s).
g) All components, software and hardware must be documented. This must include instructions on the operating system, a language manual and setup and maintenance instructions of the system.

Only those systems which met these specifications were considered. This definition also helped vendors identify which features are required in a system for educational purposes.

In looking at software features, the task force considered both whether or not a system had a particular feature as well as how important that feature was to users. Over fifty software features were identified and classified as to importance for two types of usage: Computer Science/programming and applications/programming.

The computer science/programming classification was defined as the use of a microcomputer system to meet the needs of secondary and post-secondary computer science instruction involving such topics as advanced programming, operating systems, compilers and assembly languages. This is a system used primarily by those who are interested in studying the computer system itself, therefore software flexibility is important.

The applications / programming classification includes using the microcomputer for running application programs as well as writing and running simple BASIC programs for problem solving in elementary and secondary schools. This mode requires the capability to run programs which generally range from 8 to 32K in core requirements. Since this system would also be used to run programs similar to the library programs on the existing timeshare system, downloading capability is highly desirable, if not essential.

The applications/programming classification includes using the microcomputer for running application programs as well as writing and running simple BASIC programs for problem solving in elementary and secondary schools.

In trying to evaluate microcomputers against these classifications of use, each feature was given a rating of essential, desirable, or not necessary for the two classifications as defined above. Each microcomputer was given a yes or no score on each software feature. By combining the importance scores with the yes-no scores, it was possible to give each microcomputer a software feature score for both classifications of use.

Since BASIC is the most often used language, the task force also attempted to evaluate the microcomputer's BASIC language features and capabilities which they deemed important. Test scripts were prepared and run on each of the fourteen systems.

Ten scripts were prepared. The scripts were divided into two catagories, those that tested BASIC language features, and those which tested performance. BASIC features scripts included sequential file handling, random access file handling, chaining, time function, string functions, matrix operations, and formatted output. The BASIC performance scripts included time required to complete computation (calculate number of primes from 1 to 2000), number of mathematical functions available; and time required to generate and sort 100 numbers.

In looking at hardware, some 40 features were defined and each system was rated against these. Some of the typical features included were: K-bytes of RAM, ROM, or PROM; available user memory; chip type; add time; availability of RS-232 interface and a real-time clock.

In addition to working with vendors, the task force evaluation of systems included a user survey which was developed and administered to teachers in the state who were using microcomputers in their classes. Questions dealt with types of usage, features of the microcomputers which were particularly desirable, and problems which were encountered with their systems. The major weaknesses of microcomputer systems currently in use seems to be the availability of CAI languages, ability to perform repetitive calculations, and the storage and movement of large data files. However, the majority of instructional computing can be accomplished quite well using microcomputers.

The major weaknesses of microcomputer systems currently in use seems to be the availability of CAI languages, ability to perform repetitive calculations, and the storage and movement of large data files.

Once done with defining needs, collecting data on microcomputer systems, and surveying current microcomputer users the task force made the following recommendations regarding microcomputers:

#1 State Contract:
One specific microcomputer system should be available to all Minnesota educationally-related agencies through a state contract.

#2 Support:
Instructional service support

Statewide, con't...

for selected microcomputers should be defined and increased to the same level as is currently available for large timeshare systems.

#3 Microcomputer Technology: MECC should continue to analyze and evaluate microcomputer hardware and software technology, and disseminate information to the Minnesota educational community.

On October 15, 1978, MECC and APPLE Computer, Inc. signed a contract for APPLE's 32K, disc-based Applesoft microcomputer system.

Educational computing service agencies must develop plans early if they are to cope with the fast-growing microcomputer industry.

MECC anticipates about 400-500 units to be sold by APPLE through MECC to educational users in Minnesota during the current year.

Educational computing service agencies must develop plans early if they are to cope with the fast-growing microcomputer industry. There is absolutely no indication of this growth trend slowing down. The fear that service agencies will no longer be needed is totally unfounded. Users will still need the software support that they have in the past. However, service agencies will have to redefine what they call "service" or "support." There are four major areas of microcomputer support that must be considered.

1. Purchase, installation, maintenance, and documentation of the system.
2. Training in system operations, use of application packages and programming languages.
3. Acquisition, conversion, development, maintenance, documentation and dissemination of applications packages.
4. Response to questions, problems, and requests regarding microcomputers.

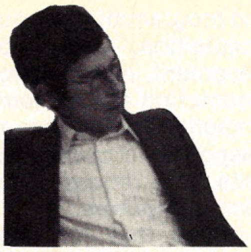
It is hoped that the comments related here will be of help to agencies faced with the problem of servicing microcomputers.

Note: A task force report was printed by MECC, and contains the Task Force research, evaluation, recommendation and the invitation for Bid for microcomputer systems. A second revision of this report will be available soon. Contact MECC, 2520 Broadway Drive, Lauderdale, Minnesota 55113 for more information. □

MECC

The Minnesota Educational Computing Consortium (MECC) was created in 1972 out of concern by the governor and legislature that educational computing needed a central source of coordination for planning, and a mechanism to insure that all educational institutions in the state would have equal opportunity of access to computing services for both instructional and administrative programs. The Consortium's membership includes the University of Minnesota (5 campuses), The Minnesota State University System (7 campuses), the Minnesota Community College System (18 campuses), the Minnesota Department of Education (representing the state's 436 independent school districts) and the Minnesota Department of Administration. Minnesota is the only state in the country having a central organization for coordinating educational computing activities across all levels of education.

The MECC Instructional Services Division offers a variety of services to consortium members. A technical staff operates the largest of Minnesota's computers dedicated to instructional computing, a Control Data CYBER 73 time-sharing system. The MECC Timeshare System is currently configured for 375 user ports and serves about 1100 interactive terminals located in schools and colleges across the state. A large multiplexing communications network provides the route by which MECC users access the Timeshare System, whether they are a few miles from the Minneapolis-St. Paul computer center or hundreds of miles away near the Canadian border. The MECC User Services staff of instructional coordinators helps users learn to make better use of the computer by visiting school and college sites, conducting workshops, providing over-the-phone consulting service, publishing newsletters and producing written documentation for programs in the MECC Timeshare System's central library.



David Levy watches the computer moves at the ACM chess tournament.

The Levy Wager

Theodore H. Ehara

*The world-famous chess champion, David Levy, has met and beaten his first computer opponent. In five years his skill is scheduled to be put to the test again with another match. Who will be the **final** Chess Master... the human, the programmer, or the computer?*

In the summer of 1968, an informal chess game was played between John McCarthy, a professor of Artificial Intelligence at Stanford University and David Levy, then Scottish Chess Champion. Upon losing the game, McCarthy remarked that Levy was the stronger player, but added that a computer program would beat Levy in a match withing the next ten years. A bet of five hundred pounds sterling (then $1,250) was made between Levy and McCarthy.

By 1975, Levy reported that he felt the two strongest programs were the Northwestern program and Kaissa, a Russian program.

During that same summer, David Slate, a graduate student in high energy physics, was walking through Northwestern University's computer center. Slate was an expert rated chess player. Although he had programmed computers to play simple games, he balked at programming a computer to play chess, because of the tremendous amount of instructions needed to obtain even a bad chess playing computer. As he walked through the systems bay area, he noticed a binder with the simple title of "CHESS" on it. The binder contained a chess playing program that was the combined work of two undergraduate students, Keith Gorlen and Larry Atkin.

Theodore H. Ehara, 1004 Hinman Ave., Evanston, IL 60202.
Photos by David Ahl.

In the years that followed that summer, Levy kept a close watch on chess playing computers, while Slate, Atkin and Gorlen (who later left the project) began to develop their mechanical chess player.

By 1975, Levy reported that he felt the two strongest programs were the Northwestern program and Kaissa, a Russian program. Levy's wager had been increased to one thousand two hundred and fifty pounds (only worth $2,500 because of the falling value of the English pound) with the addition of three other scientists betting against Levy.

Then in 1977, the Atkin/Slate program named Chess 4.5 won the Minnesota Open against human opponents. Levy played its successor, Chess 4.6, in the first formal challenge for the wager and won. At the end of that year, Levy played Kaissa and won.

As the deadline of August 1978 approached, arrangements were made for the final match between the Northwestern program and Levy to be played at the Canadian National exhibition in Toronto. Slate had already begun writing Chess 5.0, which would be a complete rewrite of the program. However, Levy played Chess 4.7 because Chess 5.0 was incomplete by the deadline.

Before leaving for Toronto, Levy received a challenge from Richard Greenblatt of M.I.T. to play M.I.T.'s MacHack. A two game match was agreed upon. Levy won both games.

In Toronto, a six game match was agreed upon. This meant Chess 4.7 needed 3½ (win 1, draw ½, loss 0) out of 6 points to win the match. Five of the six scheduled games were played. The result was Chess 4.7-1½, Levy-3½, confirming the pre-match doubts of Slate and Atkin. Figures 1 through 5 list the moves of the 5 games and board layouts at particular stages of the games.

Levy reported that by game four, he led 2½-½, he decided to try out-analyzing Chess 4.7. This accounts for the change in opening style, the sharp tactical game and Levy's only loss.

The following is an interview between **Creative Computing** and David Slate.

CC: The machine drew the first game and won the fourth game. How much of that do you think is due to Levy's knowledge of computers?

DS: Well, he knows something about the machine, and he did take advantage of the machine's weaknesses. Perhaps he did it somewhat better than another player of that rank. Actually, the machine's performance in that match was 2160 level, which is higher than its rating. I don't know what its current published rating is, but it's right around 2030. (Both ratings are expert.) That is an established rating, based on thirty-one games. So its playing strength is somewhat higher in this match than it usually is. It's not necessarily the case that Levy did so well. Rather, in the games that he won, he took advantage of what he knew were certain weaknesses of the computer - certain strategic weaknesses.

CC: Such as?

DS: When the computer is on the white side of the Sicilian Defense, he should normally get his knight on Queen four. Well, he (chess 4.7) makes the mistake of trading his knight for black's knight on Queen bishop six, which allows black to centralize (See Fig. 2) his pawns. That's something we first noticed when we played a match game with Levy on 4.5, a year and a half ago. I analyzed it and there were a number of reasons it liked to trade the knights and it wasn't so easy to prevent it. So with 4.6, there was only a small change. I was not able to make that much of a change.

He knows something about the machine, and he did take advantage of the machine's weaknesses.

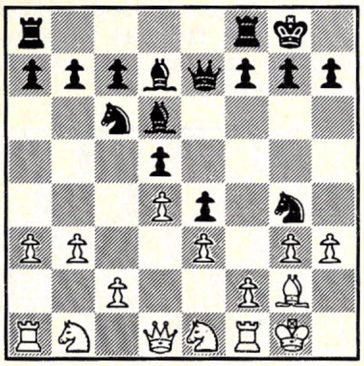
CC: So he knew about the weakness before hand?

DS: Yes he did, and he succeeded in exploiting it, in at least two games.

CC: I thought that your opening theory was kept in libraries and played by rote.

DS: Yes, it does, but you see there are many different ways you can reach

Wager, con't....



Game 1 **Reverse Pirc** Levy - CHESS 4.7

1. P-KN3, P-Q4
2. B-N2, P-K4
3. P-Q3, N-KB3
4. N-KB3, N-B3
5. O-O, B-Q2
6. P-N3, B-QB4
7. B-N2, Q-K2
8. P-QR3, P-K5
9. N-K1, O-O
10. P-Q4, B-Q3
11. P-K3, N-KN5
12. P-R3, (See fig. 1) NxP(6)
13. PxN, Q-N4
14. P-KN4, QxP +
15. R-B2, B-N6
16. Q-K2, QxR +
17. QxQ, BxQ +
18. KxB, P-B4
19. PxP, N-K2
20. P-B4, RxP +
21. K-N1, P-B3
22. N-QB3, R-R4
23. K-R2, R-KB1
24. N-Q1, N-N3
25. R-B1, BxP
26. BxB, R-B8
27. N-N2, R-B6
28. PxP, R(4)xB +
29. K-N1, PxP
30. R-B8 +, N-B1
31. B-B3, R-Q6
32. N(1)-K3, R(R)xN
33. NxR, RxN
34. B-N4, R-KB6
35. R-Q8, P-KR3
36. RxP, RxP
37. R-Q8, R-KB6
38. R-R8, P-KN4
39. P-Q5, P-KR4
40. P-Q6, K-N2
41. RxP, R-B2
42. R-R5, K-B3
43. B-B3 +, K-N3
44. R-K5, R-B6
45. B-N4, R-B5
46. R-K7, R-B2
47. RxP(4), R-Q2
48. R-K7, P-R5
49. K-N2, P-N5
50. K-R2, P-N3
51. K-N2, R-Q1
52. P-R4, N-Q2
53. P-R5, N-B3
54. PxP, N-Q4
55. P-N7, NxR
56. PxN, R-KR1
57. B-Q6, K-B3
58. P-N8 = Q, RxQ
59. BxR, KxP
60. B-B4, K-B3
61. B-Q2, K-N3
62. B-K1, K-N4
63. B-B2, K-R4
64. B-K1

Drawn

Figure 1

those sorts of positions. He would kick the machine out of the book with pawn to rook three, or something. You can't cover everything. It has to be done more generally. We managed. During the last games, I changed the openings to get out of that and we did partially succeed. The machine still lost, but it was a decidedly different kind of game.

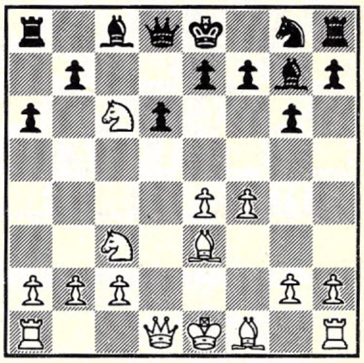
CC: How much does it cost for an average tournament game in computer time?

At commercial rates, each game would cost fifty thousand dollars, at least. But non-commercial, they do quote some in-house figure, which is not nearly that much, a few hundred dollars.

DS: Well, research and development is done on Northwestern's 660, but for tournament and matches, we tend to be on a CDC 176, because of its speed. An average tournament game, since the machine thinks on the opponent's time, essentially we're dedicating the machine to it. So if the game goes in real-time four hours, which is quite possible, it gets expensive. But we don't pay commercial rates. We're not on a production machine. We use machines in the Control Data plant that are either their in-house test machines or ones that they're testing either to ship out to a customer. So they're somewhat experimental. Sometimes that hurts us because sometimes they aren't quite reliable yet. They're still getting the bugs shaken out of them. We had some problems in the Levy match, the machine failed more than once. It failed in three of the five games. And in the last game it failed a few times. We've had some better luck. Some other tournaments, the machine has run flawlessly for five straight games. We can't predict how many times it will fail. At commercial rates, each game would cost fifty thousand dollars, at least. But non-commercial, they do quote some in-house figure, which is not nearly that much, a few hundred dollars.

CC: What about Chess 5.0?

DS: I'm working on the new chess program, Chess 5.0, now. Its a complete revision, but it shares many of the features of Chess 4.7, but it is a complete revision, in the same way Chess 4.0 changed from Chess 3.0. Chess 4.7 is written in assembly language, while Chess 5.0 is written in Fortran, a special dialect of Fortran. There's always a battle between having transportability, elegance and the features of a higher level language in computing which lets you express things simply and elegantly, which if you want to express a lot of chess ideas, you need some concise way to do that. On the other hand, higher level languages have the drawback that, unless they are very cleverly written, the programs are much less efficient than assembly language. They run much slower to do the same things. Therefore, we have a trade off. Now I'm trying to write a particular dialect of Fortran which gets the best of both worlds. It's very



Game 2 **Sicilian Defense** CHESS 4.7 - Levy

1. N-QB3, P-QB
2. P-K4, N-QB3
3. P-B4, P-QR3
4. N-B3, P-KN3
5. P-Q4, PxP
6. NxP, B-N2
7. B-K3, P-Q3
8. NxN (See Fig. 2), PxN
9. B-K2, R-N1
10. Q-B1, Q-R4
11. B-Q2, Q-N3
12. N-R4, Q-R2
13. N-B3, B-Q5
14. N-Q1, N-B3
15. P-B3, B-N3
16. Q-B2, N-N5
17. Q-R4, O-O
18. BxN, BxB
19. QxP(B), BxN
20. KxB, B-K6
21. P-QN3, BxB
22. KxB, R(N)-B1
23. Q-R4, Q-B7 +
24. K-Q3, QxP(N)
25. Q-Q4, Q-B6 +
26. K-B2, Q-K7 +
27. K-B1, P-K4
28. PxP, PxP
29. QxP, R(KB)-K1
30. Q-N3, RxP(5)
31. Q-R3, R-Q1
32. Q-B1, Q-Q7 +
33. K-N1, R-K7
34. QxR, QxQ
35. R-K1, QxR +
36. K-N2, R-Q7 +
37. K-R3, QxR

White resigns 17 moves later

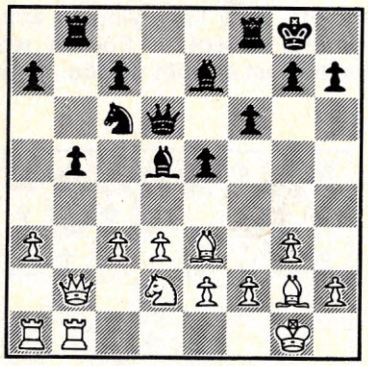
Figure 2

Wager, con't....

efficient, but at the same time it gives you a higher level language.

CC: During your match with Levy, the computer used an electronic chess board that will later be attached to a robot arm. Have you finished work on the arm?

DS: No, we're still working on it. The arm will be a very interesting addition. It will pick the piece up, slap the chess clock and shake your hand when the game is over.



Game 3 **English Opening** Levy - CHESS 4.7

1. P-QB4, N-KB3
2. P-QR3, N-B3
3. N-QB3, P-Q4
4. PxP, NxP
5. P-Q3, NxN
6. PxN, P-K4
7. P-N3, B-K2
8. B-KN2, Q-Q3
9. N-B3, B-K3
10. O-O, O-O
11. Q-R4, Q-B4
12. B-Q2, P-QN4
13. Q-B2, P-B3
14. R(B)-N1, R(R)-Q1
15. Q-N2, R(Q)-N1
16. B-K3, Q-Q3
17. N-Q2, B-Q4 (See Fig. 3)
18. BxB +, QxB
19. Q-N3, QxQ
20. NXQ, P-B4
21. B-B5, B-Q3
22. R-N2, K-R1
23. R(1)-N1, P-QR3
24. BxB, PxB
25. N-Q2, P-B5
26. K-N2, PxP
27. P(R)xP, R(N)-Q1
28. P-R4, N-R2
29. N-K4, PxP
30. R-N6, P-Q4
31. N-B5, N-N4
32. NxP(4), R-R1
33. P-QB4, PxP
34. PxP, N-Q5
35. P-K3, N-B6
36. P-B5, N-N4
37. P-B6, N-K5
38. P-B7, RxP +
39. K-N1, R(7)-B1
40. R-N8, P-KR4
41. RxR(R), RxR
42. R-N8 +, resigns

Figure 3

Like the title of Woody Allen's movie, David Levy should have been satisfied with his success and "Taken the money and run." Instead, Levy has announced that he will have a new wager with the total bet up to $10,000 in units of $1,000 with personal acquaintances. The deadline for this new wager is January 1, 1984.

In agreement with Ken Thompson, a programmer for BELLE at Bell Telephone Labs in New Jersey, Levy reported that he felt there was a barrier at the 2200 mark (the division between expert and master). Levy added that he felt the 50,000 fold increase in computing speed might be possible within the decade, which would give the ratings of the strongest programs a jump in ratings up to 2300 or above.

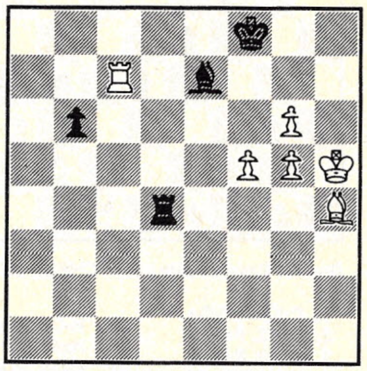
Levy feels that an advancement in hardware is possible to increase the strength of chess playing computers. He doesn't seem to feel there will be a breakthrough in the software aspects of chess computers.

However, Levy also noted that concept formation in Artificial Intelligence (AI) is one area where little or no progress has been made. So although a computer can calculate moves as fast or faster than a human chess master, it will not understand the positional ideas (like a weaken king side, stopping a flank attack, etc.) which are crucial to the game.

In the terms of a human chess player, Mr. Levy appears to have "hung his face."

Levy feels that an advancement in hardware is possible to increase the strength of chess playing computers. He doesn't seem to feel there will be a breakthrough in the software aspects of chess computers.

While Levy is looking at the history of, or lack of, concept formation in AI, it's puzzling why he feels this trend will continue. The problems of developing a chess playing computer has immediate appeal to a wide number of programmers and the entire computer field. This number is growing. It appears that Levy is not the only person who has realized this lack of concepts in programming and with the increase in people attacking this area, the chances for a breakthrough in software becomes better as the years go on.



Game 4 **Latvian Gambit** CHESS 4.7 - Levy

1. P-K4, P-K4
2. N-KB3, P-KB4
3. PxP, P-K5
4. N-K5, N-KB3
5. N-N4, P-Q4
6. NxN +, QxN
7. Q-R5 +, Q-B2
8. QxQ +, KxQ
9. N-B3, P-B3
10. P-Q3, PxP
11. BxP, N-Q2
12. B-KB4, N-B4
13. P-KN4, NxB +
14. PxN, B-B4
15. O-O, P-KR4
16. N-R4, B-Q5
17. B-K3, B-K4
18. P-Q4, B-Q3
19. P-KR3, P-QN3
20. R(B)-K1, B-Q2
21. N-B3, PxP
22. PxP, R-R5
23. P-B3, R(1)-R1
24. K-B1, B-N6
25. R-K2, B-B1
26. K-N2, B-Q3
27. B-N1, R-R6
28. R(1)-K1, R-N6 +
29. K-B2, R(1)-R6
30. R-K3, B-R3
31. N-K2, BxN
32. R(1)xB, P-B4
33. P-B4, RxR
34. RxR, R-R5
35. K-N3, R-R8
36. B-B2, R-Q8
37. R-R3, PxP
38. RxP +, K-B1
39. R-Q7, R-Q6 +
40. K-N2, B-B4
41. RxP(5), R-Q7
42. P-N4, BxP
43. R-Q8 +, K-B2
44. R-Q7 +, K-B1
45. RxP(4), R-N7
46. K-B3, B-B4
47. R-Q8 +, K-K2
48. B-R4 +, K-B2
49. P-N5, P-N3
50. R-Q7 +, K-B1
51. PxP, RxP
52. P-B5, R-R6 +
53. K-N4, R-R5 +
54. K-R5, R-Q5
55. R-QB7, B-K2 (See Fig. 4)

Black resigns

Figure 4

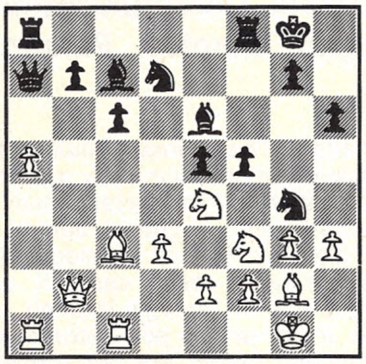
So when the deadline for Levy's new wager arrives, we can be sure of several things.

A. The programs he will be playing will be stronger.

B. Mr. Levy will be a few years older.

The fact that Levy will be older may or may not have some bearing on the wager. There has been shown a direct correlation between age and

playing strength among top players. A weaker playing skill seems to come with increased age. Still, this rule cannot be applied to a specific



Game 5 **English Opening** Levy - CHESS 4.7

1. P-QB4, N-KB3
2. P-QR3, P-B3
3. P-Q3, P-Q4
4. Q-B2, PxP
5. QxP, P-K4
6. N-KB3, B-Q3
7. P-KN3, B-K3
8. Q-B2, N(1)-Q2
9. B-N2, O-O
10. O-O, Q-N3
11. N(1)-Q2, Q-B4
12. Q-N1, P-KR3
13. P-QN4, Q-N4
14. Q-B2, N-N3
15. B-N2, P-QR4
16. P-QR4, Q-R3
17. PxP, QxP(4)
18. B-B3, Q-B4
19. R(B)-B1, N(N)-Q2
20. P-R5, Q-R2
21. Q-N2, N-N5
22. N-K4, B-B2
23. P-R3, P-KB4 (See Fig. 5)
24. PxN, PxN
25. PxP, BxP(5)
26. B-K1, N-B4
27. R(B)-N1, R(R)-K1
28. B-Q2, R-B2
29. B-K3, B-Q3
30. Q-B2, BxN
31. BxB, R-R1
32. R-QB1, P-QN3
33. K-N2, Q-N2
34. PxP, RxR
35. RxR, N-K3
36. R-R7, Q-B1
37. Q-R2, R-B3
38. R-R8, B-N1
39. B-N4, K-B2
40. Q-R7 + , BxQ
41. RxQ, BxP
42. BxN + , RxB
43. BxB, resigns

Figure 5

individual and Levy is still a young man.

In concluding this article, it should be noted that **"64"** (a Russian chess publication) related a conversation between former world champion Robert Fischer and Alfreddo Sheppeldt, during Fischer's stopover visit in West Berlin.

Fischer became attracted to a chess computer at a used bookstore he was visiting. The article did not state if he played it or not. In conversation with Alfreddo Sheppeldt, he held the belief that chess computers have a great future. □

World Checker Champion, Dr. Marion F. Tinseley has a $5,000 wager that he will beat any computer within a five year period. For further information contact the American Checker Federation, 3475 Belmont Ave., Baton Rouge, LA 70808.

If you want your chess playing program to play in a United States Chess Federation rated tournament, your program must be evaluated by the USCF Rating Committee. Request "Procedures for Computer Chess Program Registration" from the United States Chess Federation, 186 Route 9W, New Windsor, NY 12550. Membership for chess playing programs cannot be bought at tournament sites.

"There is a possibility that some existing FCC regulations on radiation or networking should be changed because they may impose a significant cost or performance penalty on users of personal computers."

The FCC Position

Remarks of Dr. Jeffrey A. Krauss, Assistant Chief, Office of Plans and Policy Federal Communications Commission before the Morgan Stanley Personal Computer Forum

Editor's Note:
The following remarks were sent to us by the Federal Communications Commission. We decided to print Dr. Krauss' remarks in their entirety because they shed some much needed light on the current controversies.

New Orleans, Louisiana, May 3, 1979.

Good morning, ladies and gentlemen, and thank you for this opportunity to discuss briefly the FCC's interest in personal computers, as I perceive it. And I want you to understand that these really are my views, and do not necessarily reflect the views of the Commission. In fact, I think it is fair to say that the FCC doesn't have any formal or official views specifically on the subject of personal computers, because the field is so new. What I hope to express is my perception of how current FCC policies of a general nature might be applied to personal computers, if the need should arise.

Now, I know that as investors and investment analysts, many of you in the audience are interested in this question: What nasty roadblocks will the government regulators throw in front of this newly-emerging technology? I hope that when I am finished speaking you will be encouraged rather than discouraged. The FCC philosophy today is moving toward less regulation — minimal regulation — and stimulating new technology rather than erecting barriers against its implementation.

This philosophy certainly reflects the views within the Office of Plans and Policy. We act as advisors to the Commission on matters of major policy significance. Our professional staff is small, about 20 people, and consists primarily of economists and engineers, with a few attorneys. The emphasis on economics and engineering reflects a view that these two disciplines have a major role to play in encouraging competition, innovation and technological change.

As I see it, the FCC has two main interests in the development of the personal computer market. First, personal computers can generate radio frequency emissions that can interfere with television reception and other radio services. Second, personal computers can be used as communications terminals attached to a communications network, enhancing the utility of electronic mail, computer time sharing and other services. Although the Office of Plans and Policy will probably have an important role in shaping the degree and nature of the Commission's relationship to the personal computer field, these two interests suggest that larger operating units within the agency, such as the Office of Science and Technology and the Common Carrier Bureau will also have a continuing role to play.

Let me first discuss RF interference, which has been a significant problem. The FCC has received a large volume of complaints about CB radio transmissions interfering with television reception. RF interference caused by personal computers is not yet a significant problem. To put it bluntly, we don't want it to become a significant problem. Based on the technical knowledge that some manufacturers already have and others are rapidly developing, I am encouraged that this will not develop into a serious problem.

Perhaps some background might be helpful. The internal circuitry of a personal computer typically operates a megahertz frequencies, and it generates radio frequency energy. If the circuitry is not shielded, for example by a metal case, it can radiate the RF energy through the case. If the various power cords and connecting cables are not properly filtered or shielded, they become antennas and conduction paths for the RF energy. There are some personal computers on the market now that apparently lack the necessary shielding and filtering. As a result, the FCC has begun to get some interference complaints.

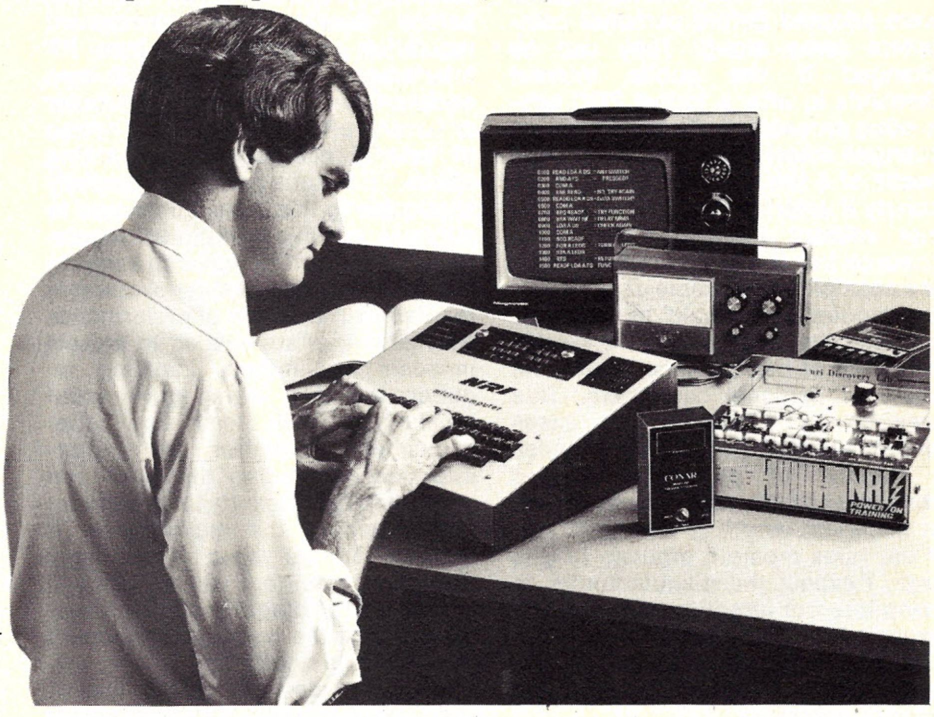
Personal computers are classified as "restricted radiation devices" subject to Section 15.7 of the FCC's Rule. This regulation was adopted in 1938 when we were basically concerned with AM broadcasting, which operates at kilohertz frequencies. The general feeling at the staff level is that the old regulation is no longer appropriate. Accordingly, the Commission started a proceeding looking toward changing the Section 15.7 standard as applied to computers. The focus of that proceeding is on commercial computers rather than personal computers, but the findings may also be applicable to personal computers.

We have another regulation pertaining to equipment that has a built-in RF modulator, such as a video game. This enables the home TV set to be used as an output display. Section 15.419 of the FCC Rules imposes the same stringent performance standards on video games — called "Class I TV devices" in our Rules—that Section 15.7 imposes on restricted radiation devices generally. In addition, the Class I TV device must be submitted to the FCC laboratories for testing and approval, while restricted radiation devices in general need not be submitted for compliance

Dr. Jeffrey A. Krauss is the Assistant Chief of the Office of Plans and Policy for the Federal Communications Commission.

Build your own microcomputer as you learn computer technology at home.

New from NRI! The Most Complete and Up-to-date Home Study Course Ever Offered



As the microprocessor revolutionizes the computer world and microcomputers appear almost everywhere, NRI brings you a new, convenient, and effective way to keep up with this expanding technology. It's NRI's Computer Technology Course, created and designed exclusively for learning at home in your spare time.

Featuring NRI's Exclusive Dual Language Microcomputer

NRI goes beyond book learning to give you practical, "hands-on" experience in designing circuitry, interfacing components, programming, and troubleshooting. As you learn, you actually assemble NRI's designed-for-learning microcomputer, incorporating the latest advances in the state of the art. It looks and operates like the finest of its kind, actually does more than many commercial units. But NRI engineers have designed components and planned assembly so it demonstrates important principles, gives you working experience in detecting and correcting problems. And it's yours to keep, put to work in your own home or business.

You also build and keep your own test instruments, including a transistorized volt-ohm meter and CMOS digital frequency counter. And NRI's Discovery Lab® broadens your horizons with specialized experiments and theory demonstrations.

The Proven Way to Learn at Home

You don't have to worry with travel, classes, or time lost from work when you learn the NRI way. As they have for more than 60 years of teaching technical subjects, NRI brings the material to you. You study in your spare time, at your convenience, using "bite-size" lessons that program material into logical segments for easier assimilation. You perform experiments and build equipment using kits we supply. And your personal NRI instructor is always available for consultation should you have questions or problems. Over a million students have already shown the effectiveness of NRI training.

Choice of Courses

Several courses are available, depending upon your needs and background. NRI's Master Course in Computer Technology starts with the fundamentals, explores basic electronics and digital theory, the total computer world, and the microcomputer. The Advanced Course, for students already versed in electronics and general computers, concentrates on the microprocessor and microcomputer. In both courses, you build all instruments and your own computer.

Send for Free Catalog… No Salesman Will Call

Get the details on these exciting new courses in NRI's free, 100-page catalog. Shows all kits and equipment, lesson outlines, and full information, including facts on other electronics courses. Mail the coupon today and we'll rush your catalog. No salesman will ever call. Keep up with the latest technology as you learn on your own computer. If coupon has been removed, write to NRI Schools, Computer Department, 3939 Wisconsin Ave., Washington, D.C. 20016.

NRI Schools
McGraw-Hill Continuing Education Center
3939 Wisconsin Avenue
Washington, D.C. 20016

NO SALESMAN WILL CALL
Please check for one free catalog only.

All career courses approved under GI Bill.
☐ Check for details.

- ☐ Computer Electronics Including Microcomputers
- ☐ TV/Audio/Video Systems Servicing
- ☐ Complete Communications Electronics with CB • FCC Licenses • Aircraft, Mobile, Marine Electronics
- ☐ CB Specialists Course
- ☐ Amateur Radio • Basic and Advanced
- ☐ Digital Electronics • Electronic Technology • Basic Electronics
- ☐ Small Engine Repair
- ☐ Electrical Appliance Servicing
- ☐ Automotive Mechanics
- ☐ Auto Air Conditioning
- ☐ Air Conditioning, Refrigeration, & Heating Including Solar Technology

Name ____________ (Please Print) ____________ Age ____

Street ______________________________

City/State/Zip ______________________________

Accredited by the Accrediting Commission of the National Home Study Council   175-099

FCC, con't....

testing. On the other hand, we do not currently have any provision in our regulations for RF modulators that stand alone, without a permanent connection to the RF signal source. As some of you may know, Texas Instruments has recently filed a petition with the FCC, asking that we establish standards for a "Class II TV device" that includes the modulator but not the RF signal source.

All of the current FCC regulations were adopted before personal computers came along. They can be changed if the public interest demands it, after a formal FCC rule-making proceeding. They will not be changed merely to meet the private needs of a few manufacturers, particularly if existing regulations can be met without imposing significant costs on consumers. However, there is a possibility that some of our existing regulations should be changed because they may impose a significant cost or performance penalty on users of personal computers. This may be particularly true in cases where the computer system includes a number of peripheral devices connected to the computer by cables of uncertain length or configuration. If our rules create a problem in this area, I expect the manufacturers to bring it to our attention, and I would expect the Commission to adopt changes if the changes are technically valid and consistent with the public interest.

While there may be changes in RF standards in the future, personal computer manufacturers cannot rely on speculation about future changes. They must design their equipment to meet the current standards. Those manufacturers who employ design engineers with backgrounds in video games, military electronics or ham radio, probably already have the necessary expertise. I encourage the others to develop this expertise.

Meanwhile, as complaints arise from the computers now in the market, I expect manufacturers to resolve the problems on a case-by-case basis. We have indications that this is occuring now on a voluntary basis. Let me quote from a recent letter to the FCC:

> (T)he computer company is attempting to solve the (interference) problem that I am having with my computer. The company has supplied me with a new cable between the key board and the motherboard. This has been fairly effective... In addition, (the company) is having a special power supply built for my computer. ...I believe that (the company) is seriously trying to solve the (interference) problems.

This reflects a responsiveness and sense of responsibility that I find very encouraging.

Before moving on, let me add another point of view to the discussion. If interference does occur, it will usually be to TV sets or other equipment in the same house. If the same person has control over both the computer and the TV set, then that person doesn't need government regulation to protect him from RF interference; he can make his own decision by turning off the computer or turning off the TV set. This notion of "interference between consenting adults," if it turns out to be generally applicable to this case, could lead to minimal, passive government regulation in the form of labeling requirements rather than active government regulation in the form of performance standards that are needed to protect innocent bystanders. Labeling requirements and other government regulations that lead to better consumer information and better-informed purchasing decisions can contribute significantly to the smooth functioning of the marketplace. It remains to be seen how many personal computers will be used in apartment buildings and other situations where the number of innocent bystanders is significant.

Now let me turn to the use of personal computers as communications terminals. The FCC has established policies in a number of decisions over the past decade of allowing, indeed, encouraging, users of the telephone network and other communications common carrier services to optimize their use of these facilities and services. In these decisions the FCC removed restrictions that communications carriers had imposed on the use of their services. In addition, the Commission has encouraged service offerings of a wide diversity and variety, through decisions allowing competition and open entry in many segments of the communications industry. Based on these trends, I feel fairly confident that the Commission would encourage the inter-connection of personal computers into communications networks.

While the networking of personal computers could result in service enhancements in a number of areas, I believe the most important areas are electronic mail, and the rapidly-developing community bulletin board services. Although electronic message services can be provided equally well by traditional computer time sharing companies, they will develop faster and be more responsive to personal needs (as opposed to commercial needs) if they are available through personal computer networks. I feel confident that all of the necessary computer processing features of an electronic message system, such as text editing and message storage and retrieval, can be as efficiently incorporated into a personal computer network as into a time sharing service.

I perceive some potential barriers to the expansion of personal computer networks, however. The rate structures of the data communication services now available do not seem to be geared to personal use, but rather are intended for commercial use. This has resulted in the use of the long distance telephone network for personal computer networking, in spite of its sometimes inadequate quality. In addition, the current price of adding communications capability on some of the personal computers is too high and will have to come down if communications use is to be encouraged. If I can buy a personal computer for $600, why must I pay $300 to add on a communications capability? Finally, there is the spectre of regulation. What happens all too often in government regulation is that service vendors use or abuse teh administrative process for their own personal short-term gain. It may be that some vendors who fear competition from personal computer networks will argue that personal computer networks should be regulated as common carriers. From what I know of the developing applications, I see no economic or technical need for FCC regulation. Competitors and their attorneys who choose to argue in favor of common carrier type regulation will have a heavy burden to demonstrate the need for regulation in today's environment.

In closing, let me say that I hope that these remarks have been more encouraging than discouraging. Too often, technological innovation has been slowed by government regulation. It is my hope that government regulation will not slow the development of personal computers. A responsible industry approach to the problem of RF interference will help to assure that government involvement is minimal, at least on the regulatory side. There may be substantial government involvement on the user side, but I'll leave that discussion to the marketing experts. Thank you. □

THE SEARCH FOR A SMALL COMPUTER SYSTEM STARTS HERE

It's the 3rd Annual National Small Computer Show

New York Coliseum, August 23-26, 1979

presenting the state-of-the-art showcase for micro-and mini-systems technology and software. Here you can survey virtually all makes and models of small computers, whether your interest runs to a no-nonsense micro priced in the hundreds of dollars or a powerful mini costing $20,000 or more. They're all here.

The world of small computers is quite large, extending to business and professional offices, scientific research, medicine and bionics, education, the home and hobbyist, therapeutic applications for the handicapped, design and engineering. A full selection of lectures is presented to provide a grasp of small systems technology, so that you know what to consider when buying a computer or word processor. It's the first step in discovering what a system can really do for you!

NSCS lectures include sessions on system selection, computer languages, word processing functions, artificial intelligence, software applications, and a dozen more topics for people of all interests.

Plan now to attend. There will be about 30,000 square feet of exhibits, and more than 40 hours of lectures from which to choose. Registration fee is only $5.00 per day, including lectures.

LECTURES: (Program subject to change)

Thursday, August 23

- **1 p.m.** The Peril of Becoming a Machine-Oriented Business User
- **1 p.m.** Introduction to Small Business Systems
- **2 p.m.** Selecting a Word Processing System
- **2 p.m.** Distributed Data Processing
- **3 p.m.** Accounts Receivable/General Ledger/Accounts Payable
- **3 p.m.** Is There a Computer in Your Educational Future
- **4 p.m.** Mailing Lists: Load, Time and Cost
- **4 p.m.** Word Processing Systems in the Law Office
- **5 p.m.** Basic BASIC
- **5 p.m.** Achieving Quality Control in Word Processing

Friday, August 24

- **1 p.m.** The Peril of Becoming a Machine-Oriented Business User
- **1 p.m.** Introduction to Small Business Users
- **2 p.m.** Selecting a Word Processing System
- **2 p.m.** Distributed Data Processing
- **3 p.m.** Unassigned at press time
- **3 p.m.** How to Write a User-Oriented Program
- **4 p.m.** Efficient Expansion of a Small System
- **4 p.m.** Investment Analysis
- **5 p.m.** Accounts Receivable/General Ledger/Accounts Payable
- **5 p.m.** Exploiting the Apple/Dow Jones Computer Link

Saturday, August 25

- **11 a.m.** Introduction to Personal Computing
- **11 a.m.** Unassigned at press time
- **12 p.m.** Computer Music Update
- **12 p.m.** Unassigned at press time
- **1 p.m.** Introduction to PASCAL
- **1 p.m.** Computer Art Forms
- **2 p.m.** Household Applications
- **2 p.m.** Artificial Intelligence
- **3 p.m.** How to Write a User-Oriented Program
- **3 p.m.** Investment Analysis
- **4 p.m.** Basic BASIC
- **4 p.m.** Unassigned at press time

Sunday, August 26

- **11 a.m.** Introduction to Personal Computing
- **11 a.m.** Computer Music Update
- **12 p.m.** Household Applications
- **12 p.m.** Unassigned at press time
- **1 p.m.** Efficient Expansion of a Small System
- **1 p.m.** Computer Art Forms
- **2 p.m.** Unassigned at press time
- **2 p.m.** Unassigned at press time
- **3 p.m.** Microcomputers for the Handicapped: Update
- **3 p.m.** Exploiting the Apple/Dow Jones Computer Link
- **4 p.m.** Mailing Lists: Load, Time and Cost
- **4 p.m.** Introduction to PASCAL

REGISTRATION FOR AMERICA'S BIGGEST SMALL COMPUTER SHOW

Please register me for the 3rd Annual National Small Computer Show, Aug. 23-26, 1979.

NAME ______________________ BUSINESS TITLE (If Any) ______________

COMPANY (If Any) ______________________ TELEPHONE ______________

ADDRESS ______________________

______________________ ZIP ________

(Check main interest)

- ☐ Banking/Insurance
- ☐ Business office
- ☐ Communications
- ☐ Educational
- ☐ Government
- ☐ Hobby
- ☐ Industrial/Manufacturing
- ☐ Military
- ☐ Professional
- ☐ Stock Brokerage
- ☐ Transportation
- ☐ Utility
- ☐ Wholesale/Retail
- ☐ Other ________

(Check main job function)

- ☐ Accountant
- ☐ Administrator (Business)
- ☐ Architect/Builder
- ☐ Art Director
- ☐ Banker
- ☐ Computer technician
- ☐ Consultant
- ☐ Controller
- ☐ Engineer
- ☐ Industrial Designer
- ☐ Lawyer/law office mgr.
- ☐ Marketing manager
- ☐ Medical doctor
- ☐ Medical technician
- ☐ Office manager
- ☐ Programmer
- ☐ Public Servant
- ☐ Research/Development
- ☐ Teacher
- ☐ DP manager
- ☐ WP manager
- ☐ WP operator
- ☐ Stock Broker
- ☐ Systems Analyst
- ☐ Student
- ☐ Other ________

☐ ONE DAY $5 ☐ TWO DAYS $10
☐ THREE DAYS $15 ☐ FOUR DAYS $20

Mail with payment of $5 for each day you wish to attend. Use one form per person. Registration badge will be sent by mail in early August. Check or money order only.

Mail prior to Aug. 10 to:
National Small Computer Show,
110 Charlotte Place,
Englewood Cliffs, N.J. 07632.

There is the danger in any large-scale program of getting into an infinite loop. Here is a theoretical method to detect such loops.

Infinite Loop Finders Revisited

W.D. Maurer

For many years I have been telling people that infinite loop finders **can** be made to work, in contrast to the conventional wisdom in computer science which says that they **cannot**. There is a species of computer-person who may be called the "knee-jerk unsolvability theorist," whose mind is closed to the consideration of any possible imperfections in the logic by which he infers invalid practical consequences from the unsolvability of the halting problem. To such people and others I reiterate: An infinite loop finder **is possible** — although very difficult.

The usual paradoxes associated with infinite loop finders involve the application of an infinite loop finder to itself or to some reasonable extension or variation of itself. These paradoxes do not necessarily hold when the source and the object machines are not taken to be the same; that is, when the infinite loop finder runs on a machine B, but finds infinite loops in programs which themselves run on some other machine A. In fact, for every machine A, I can exhibit a machine B and a relatively simple and straightforward infinite loop finder, which runs on B, and which finds infinite loops in programs which run on A. The only restriction on machine A is that it be constructible in the real world.

The argument goes as follows. We initially assume that the machine A is deterministic; that is, each state of the machine A gives rise to one and only one next state. We also initially assume that A has no tapes or other input-output. In that case, suppose that the total number of bits in A is **n** — that is, the total number of bytes is **n/8**. (These include, however, all registers, status flags, program counters, etc., whether program-accessible or not.) The machine B has total number of bits **3n** + 2^n + **k**, where the 3n bits are three copies of A, called M, L, and J (each of which can hold any integer value from 0 to 2^n-1); the 2^n bits form a bit array T of dimension 2^n; and k is the number of bits necessary to encode the instructions of the infinite loop finder. This works as follows, assuming that it is trying to find an infinite loop in the program P which runs on machine A (the program P, together with an interpreter for machine A which runs on machine B, must be contained in the **k** bits mentioned above):

1. For M = 0 to 2^n-1, do steps 2 through 11 below.
2. (Find an infinite loop if program P is started in machine state M.) If M is not a legal configuration of machine A for program P (that is, if the instruction words for the instructions of P are not all in their proper places), go to step 11 below.
3. For L = 0 to 2^n-1, set T(L) = 0. (T(L) = 1 will mean that the L-th state of the machine is one of the states which the machine A has been in during this particular simulated run.)
4. Set T(M) = 1. (Start in the M-th state.)
5. Set L = M. (L is the current state, initially the M-th state.)
6. For J = 1 to 2^n-1, do steps 7 through 9 below.
7. Interpretively execute one step of the program P (assuming that L represents the current state of the machine A), and set L equal to the resulting state of the machine A.
8. If now T(L) = 1, stop; there is an infinite loop in P (because the state of the machine A is exactly as it was at some previous time; if it took x steps to get to this state initially, and then y steps to get to it the second time, it will re-enter this state at step x + 2y, X + 3y, and so on, and thus will cycle indefinitely, since it is deterministic and cannot be affected by the external world). Otherwise, set T(L) = 1.
9. If the next instruction of P is a halt instruction, go to step 11 (no infinite loop found if M was the initial state).
10. Stop. (At this point we have cycled through all 2^n states of A. The next state of A must be one that we have seen before, so that there is an infinite loop in P, just as at step 8).
11. Continue (do nothing; end of outermost loop).
12. At this point we know that there is no infinite loop in P, because we have run P for all possible inputs, and every time we have run P it has halted.

Let us now drop the requirement that A be deterministic. In this case we use the same construction as above, except that the bit array T now has dimension 2^{2^n}, rather than 2^n, and each T(J) is indexed by an arbitrary **subset** J of the set of all states of A. Also, M, L, and J can now take values from 0 to 2^{2^n}-1. If J is the current subset of states, we obtain the next subset of states by interpretively executing one step of the program P for each state in J, and then considering the union of all sets of next states obtained in this way.

Next, suppose that machine A has tapes or other input-output devices which can be read or written, but does not accept a potentially infinite stream of input data from an external source. The point is that the total capacity of the tapes or similar such devices must be be finite. (This is true for physical reasons: the world contains only a finite number of atoms, and hence only a finite number of tapes). Thus again we may calculate the total number **n** of bits in A, this time including bits recordable on the various tapes or other such devices, and so this case reduces to the previous ones.

Finally suppose that A accepts a potentially infinite streat of input data. Although the amount of matter in the world is limited, the amount of time is not; one can imagine a computer accepting an abrbitrary stream of input without limit, so that the "total capacity" argument given above does not apply. However, this case can be neatly reduced to the preceding ones by considering any input instruction as a nondeterministic instruction, in which the register which receives the input can hold an arbitrary input value in the next state after the input instruction is executed. This argument takes advantage of the fact that the input data is truly **input** data only and is not data that was previously written by this

W.D. Maurer, Professor, George Washington University, S.E.A.S., Washington, D.C. 20052.

machine on a tape, for example (which is the preceding case).

Of course, we have yet to consider the problem of finding infinite loops in a program that runs on machine B (such as the preceding infinite loop finder). But this is easy enough; we merely apply the same process to B as we applied previously to A. This process produces a machine (call it C) which bears the same relationship to B as B does to A. In fact, given any machine A, we can produce in this way an infinite sequence of machines $A_1, A_2, A_3, \ldots$, with A_1, = A, and an infinite sequence of loop finders $L_1, L_2, L_3, \ldots$, such that each L_i, finds infinite loops in any program running on machine A_i, and each L_i itself runs on machine A_{i+1}. The process by which A_{i+1} is derived from A_i, for each **i**, is exactly the same as that by which B was derived from A.

Now that this construction is complete, what does it have to do with the unsolvability of the halting problem? The answer is actually quite simple. When Turing constructed his mathematical model of a computer — which has since come to be called the Turing machine model — he constructed it in such a way as to have an infinite tape. Obviously, this is not because Turing really believed in an infinite number of atoms in the world. It was simply because the problems which arise when a machine runs out of tape are seemingly irrelevant to the basic questions having to do with the nature of computation. If a machine can calculate a function, but in practice does not because it runs out of tape, we would still like to be able to regard that function as calculable by that machine. However, given this reasonable feature of the Turing machine model, the unsolvability of the halting problem is a very curious theorem: its truth depends on having an infinite tape, and it ceases to be true whenever the tape is finite, however large. This, of course, is one of the dangers inherent in any kind of modelling, although it is much better understood in connection with econometric models. If one makes a model of the economy, which necessarily is only an approximation to the real world, do the mathematical properties of that model always hold in the real world? Of course they don't; and the reason why is that, just as in the case of Turing machines, we have a property of a model that applies to the model but not to the situation which it is (approximately) modelling.

The solvability of the halting problem for finite machines has actually been commented on by a number of unsolvability theorists, usually followed by the statement that the halting problem for finite machines is difficult. This is perfectly true; in the deterministic case, if the machine B, as above, has 16 megabytes (say), the machine A can have a total capacity of only 24 bits. Any reasonable machine A gives rise, using this construction, to a machine B which is far too large to be constructed in practice (even using all the atoms in the world). A much more interesting question, however, concerns **practical** infinite loop finders. Such a program would be able to find (let us say) 65% of all infinite loops that occur in practice, and it would be able to do this because of its knowledge that there is a collection of common mistakes that programmers make. The infinite loop finder searches for instances of these common mistakes, finds them, and corrects them. If it cannot find an infinite loop, it simply prints out CANNOT FIND INFINITE LOOP (instead of NO INFINITE LOOP, which would not necessarily hold).

Such an infinite loop finder would be similar to programs which correct errors on tapes (obviously, it is impossible to correct **all** tape errors through a single program), as well as to other programs which are known, in the world of artificial intelligence, as heuristic programs — they sometimes report inability to complete their assigned task. The knee-jerk unsolvability theorists, however, have effectively prevented such work from even being started. Or have they? Is there somewhere, out in the world, a programmer who has written a practical infinite loop finder? If so, let me hear from you! □

"When I got it about half completed, it started criticizing the way I was putting it together!" ©Creative Computing

creative computing

Vol. 2, No. 5—Sep/Oct 1976
Computer programming contests, Russian computing, Do-it-yourself computer poetry (3 articles), two games: Watchman and Delmar, four feature reviews of "Computer Power and Human Reason," Computers in elections, two great stories.

Vol. 3, No. 1—Jan/Feb 1977
Equipment profiles: Teletype model 43, IMSAI 8080, SWTPC 6800, Computers in the movies, All about Electronic Funds Transfer, Centerfold "Computer Tree" Babbage to 370/158, A approach for analyzing discontinuous events, unsolvable complex problems, the Wooly Mammoth problem, ten outstanding problems for computer solution. Games: Drag Masterbagels, Strike 9.

Vol. 3, No. 4 - Jul/Aug 1977
Guide to selecting a microcomputer. Write your own CAL, Part 2. Computers in medicine and health care. Dwyer: "8-Hour Course in Basic-Part 1." Thinking Strategies-Part 3." Sherlock Holmes and Charles Babbage. Four new games.

Vol. 3, No. 5—Sep/Oct 1977
A dynamic debugging system for 8080 assembly language, bibliography of "limits to growth" models, Dywer: 8-hour course in Basic-Part 2, Programming approaches to solving complex equations, Electronic information exchange, Symmetric art with your computer, in-depth reviews of 5 microcomputer BASICs, software technology music system, Games: Nomad, Rotate, Lissajous.

Vol 3, No. 6—Nov/Dec 1977
Programming techniques- Part 1. CAI. Topics in Logic. Three 8080 8K BASIC evaluations. Smart electronic game reviews. How computers can write final exams. Mastermind II and Othello computer games. Profile of the Alpha 1 and Alpha 2 for the TDL Xitan.

Vol. 4, No. 1—Jan/Feb 1978
File structures, 16-bit computers, LOGO Language, Murphy's laws, review of Radio Shack TRS-80 and Heath H8, World model, biorythms, how to write a simulation, Hart sort algorithm, 3 games, 8-Hour Basic Course - Part 4.

Vol. 4, No. 2—Mar/Apr 1978
Parody of *Datamation*, Business Computing: 5 inventory control systems, ABCs of microcomputers, structured software for micros, four computer music systems, reviews of 2 Basic interpreters and micro-APL, CAI-Part 4, puzzles and games.

Vol. 4, No. 3—May/Jun 1978
Art and animation section: 8 articles, color graphics, SAM76, binary search, a real budget in Basic, business computing: 4 payroll systems, Oregon Trail, Black Box, reviews of VideoBrain, MSI floppy, OSI Challenger, Ai speech synthesizer.

Vol. 4, No. 4—Jul/Aug 1978
Reviews of Commodore PET, Apple II, Atari computer, Video games, interfacing to the real world: 5 articles, business computing: 4 word processing systems, ROM section: 7 articles, backgammon game, bar code.

Vol. 4, No. 5—Sep/Oct 1978
Equipment profiles: TRS-80, Exidy sorcerer, Bally Arcade, PolyMorphic 8813, Merlin Video Display preview of nine new personal computers. Accounts receivable systems, All about PASCAL, real world games, a real time clock to build, PET cassettes, special education features, new software: Star Wars, Hex.

Vol. 4, No. 6—Nov/Dec 1978
Subject index and file index in BASIC, consumer computers buying guide, electronic game reviews, critical path analysis, mailing label programs, robot programming, experiment in teaching strategic thinking, evaluations of Northstar Horizon, CP/M operating system and backgammon computers, columns on Apple II PET and TRS-80, plus game section including "Corral", "Joust" and Puzzle"

Vol. 5, No. 1—January 1979
Computers in fiction; Survey of Educator's Attitudes; K-State; How to Hide Your Basic Program; World Chess Championship Computer; Compleat Computer Catalog, Microchess for the TRS-80; Exidy Sorcerer; Ohio Scientific superboard II; Robots in Fiction; Help for the Weary Taxpayer; A counterfeit Cursor for your PET; Medical Audit Time.

Vol. 5, No. 2—February 1979
Evaluations: Electric Pencil, Heathkit H-8, Computer Music Records. Computer Games: Gold Mine, Atom-20. Computerized Sports Predictions, Multiple Regression Analysis Simplified, Value of Computers in Education, Budget Management System, Help for the beleaguered consumer.

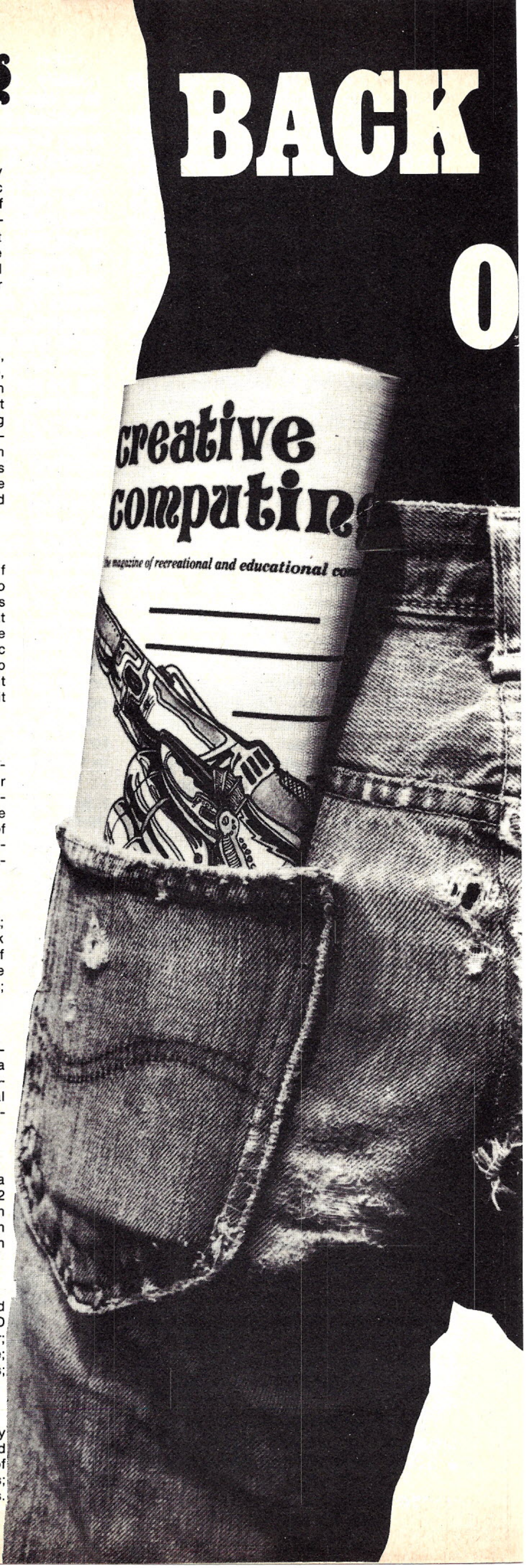
Vol. 5, No. 3—March 1979
Six articles on data base management; Evaluations of TRS-80 and Apple Disk Systems; Payroll system; the Game of Go; Small business computing with the Sourcerer; Judging of sports events; Social Science survey program.

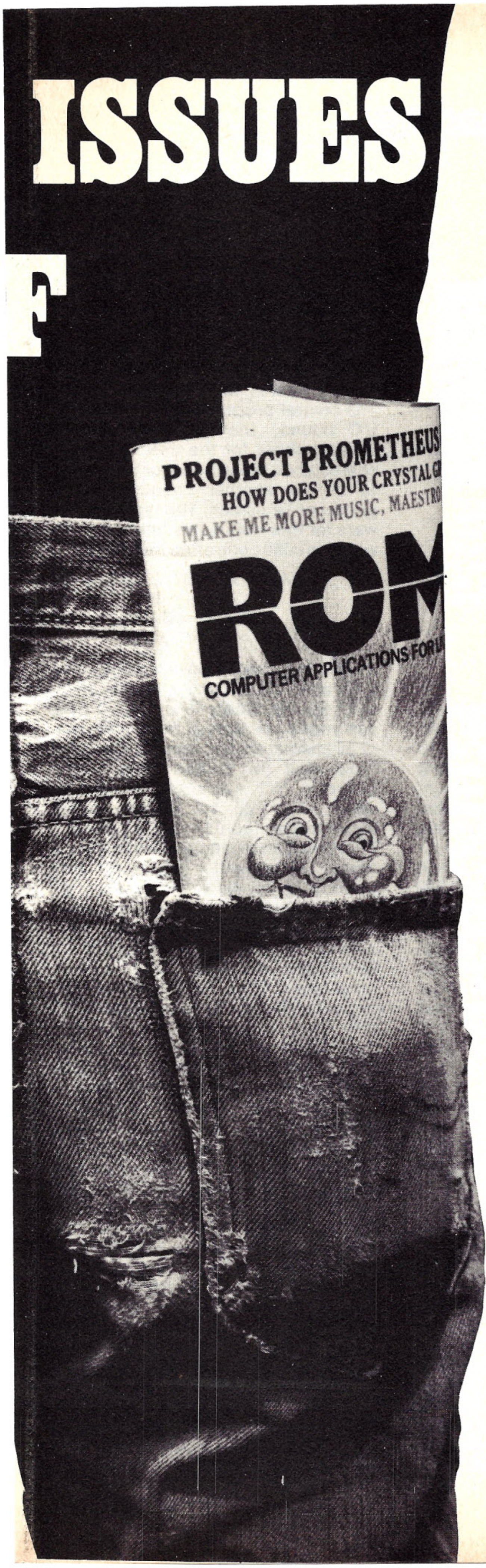
Vol. 5, No. 4—April 1979
Safeguarding your computer; Interpretive programming; Elements of a good computer game, Music composition; "What will happen if"; Vertical graphs and bar charts; People Programming; Home applications.

Vol. 5, No. 5—May 1979
Word processing systems — buying a system and 5 evaluations; Writing 2 user-oriented program; Tutorial on PILOT; 3 new games; Amoritization schedules, reading and comprehension tests.

Vol. 5, No. 6—June 1979
8 Articles on computer graphics and plotting; Evaluations: HiPlot, NAD System, ALF/Apple Music Synthesizer; Copyright of Software; Sesame Place; Probability Study; String Manipulations; 3 New Games.

Vol. 5, No. 7-July 1979
Two Ecology Simulations, Creativity Test; World Power Systems; Files and Data Basis — 4 Articles; Evaluations of Six Peripherals and Software Systems; Personal finance Model, 2 logic games.





Are you missing any back issues of **Creative Computing** or **ROM** magazine? The applications, programming techniques, simulations, problems, commentary, articles and fiction are practically timeless. Not only that, but the earlier issues are actually increasing in value.

Prices are $2.00 each, three for $5.00, or ten for $15.00. Postage $1.00 for up to 3 issues, $2.00 for 4 or more.

SUPER SPECIAL: One of everything we have plus 4 back issues of **Computer Notes** — 32 magazines in all — for only $40 postpaid!

ROM

July 1977
SOL. The Inside Story; Braille and the Computer Video newspaper; A Chip is Born; The Care and Feeding of Your Home Computer; Digital Foam — the peripheral of the future.

August 1977
The Kit and I, Part I, by someone who's never soldered before; Introduction to the fundamentals of Computer Memory; Tips for the do-it-yourself hardware beginner; Binary clocks; APLomania.

September 1977
Xeroxes and other hard copy off your CRT; Payroll Program; How Computers Work; The Kit and I, Part II: or Power to the Computer; CCD's How They Work and How They're Made; A look at PLATO, an Educational Computer System; IBM 5100.

October 1977
Binary Arithmetic For the Beginner; Microprocessor Aid for the Deaf and Blind; The Kilobyte Card; Scott Joplin on Your Sci-Fi Hi-Fi; Building a Basic Music Board; Flowcharting; Payroll Program.

November 1977
Solar Energy Measurement; A Beginners Introduction to BASIC; The Kit and I, Part III; More Music to Play on Your Computer; Micro Maintenance; Solomon and Viet: Putting Together a Personal Computing System; Time Sharing on the Family MICRO.

December 1977
A Beginners Guide to Peripherals; The Best Slot Machine Game ever; Artificial Intelligence?; An Electronic Jungle Gym for Kids; File Copy Program; Better Health Through Electronics; the Kit and I Part IV.

January 1978
Synthetic Skin for Your Robot and How to Make It; TLC: A Visual Programming Language; The Code That Can't Be Cracked; Beginners Guide to Computer Graphics; The Computer and Natural Language; First-Timer's Guide to Circuit Board Etching.

February 1978
A Practical Mailing List Program; Artificial Intelligence; Assemblers; Flowgrams—A New Programming Tool; Refresher Course in BASIC; Micros and Analyzing Election Results; Upgrading Your BASIC.

March-April 1978
Introduction to real time concepts; Felsenstein: An Absolute-Time Clock; Dreyfus: Things Computers *Still* Can't Do; Introduction to Interpreters; Othello Games; Weizenbaum: Incomprehensible Programs; The Quasar Robot Revealed; Chesson: Cryptanalysis.

December 1977
A Beginners Guide to Peripherals; The Best Slot Machine Game ever; Artificial Intelligence?; An Electronic Jungle Gym for Kids; File Copy Program; Better Health Through Electronics; the Kit and I Part IV.

Send order to Creative Computing, P.O. Box 789-M, Morristown, NJ 07960. Or save time and call your credit card order toll free to: **800-631-8112** (in NJ, 201-540-0445).

Please rush me the following back issues:
Magazine and Issue

____________________ ____________________
____________________ ____________________
____________________ ____________________
____________________ ____________________

Amount for mags: ________ + Postage ________ = Total ________

☐ Payment enclosed ☐ Visa/Bank Americard ☐ MasterCharge ☐ American Express

Card No. ____________ Expiration date ________ Signature ____________

Name ____________________________
Address ____________________________
City ____________ State ________ Zip ________

Micro Time-Sharing

Jeff Levinsky

During the past year and a half a group of us at Clairemont High School in San Diego, CA., has developed several time-shared systems on 8080-based microcomputers. These systems provide multiuser interactive capabilities at very low cost compared to commercial time-shares (which typically require mini-computers and special memory configurations) and are useful in educational, business and hobbyist applications. At present, we run up to six terminals simultaneously on an IMSAI with a student-written program called CHATS (Clairemont High Altair Time Share) which time-shares MITS Extended BASIC (version 3.2). CHATS "patches" into that BASIC and gives each user the full interactive capabilities of the language. Our current efforts are directed towards a more ambitious system called CHAOS (Clairemont High Altair Operating System) which not only provides a time-shared version of MITS Disk BASIC but also highly flexible account and command manipulation mechanisms and several other languages. We believe, as a result of our successes in these lines, that microcomputers can and should be used for time-sharing and can rival commercial systems in price, performance and power.

The heart of the CHAOS II system; an Imsai 8080 with MITS floppy disk system. The Imsai contains 48K Extensy's RAM, (2) 8K Processor Tech RAM boards, I Cromemco Bytesaver, 3 MITS 2 S/O boards (for 6 ports), (2) 3P + S Processor Tech I/O boards and Tarbell Tape Interface.

Why Time-Sharing

We were originally drawn to time-sharing as an economical alternative to our district's Hewlett-Packard 2000 Access BASIC system. In 1975, our school purchased a port and terminal on this system for $7500. Maintenance fees are an additional $500 per year. Since we own a share of the computer, we have unlimited time at no extra charge and about 256K 16-bit words of memory. The HP system does provide relatively good interactive programming and file storage capabilities but a single port is inadequate to meet the demands for terminal time by many interested students, teachers and administrators. To conduct a class in computer mathematics or computer science (in which there is heavy emphasis on programming exercises) we felt that the school must have at least six to eight terminals to give every student sufficient on-line time. The cost of additional ports and of the associated maintenance on the HP system made expansion in that direction impossible. We knew at that time of no alternative system for under $10,000 and so we decided to build our own. Subsequently, we purchased an ALTAIR 8080, some memory, several SWTPC TV terminals, and the related input/output hardware and set about modifying already existing one-user programs, such as games and BASICs, into multiuser ones. Success at these projects lead to further purchases, including much more memory, two Processor Technology SOLs, and IMSAI, a SWTPC line printer, and a soon to arrive Diablo printer. While a five-user operation on the district's HP would cost us $37,500 plus $2500 per year (and many schools will pay this), a comparable 8080-based time-share might consist of: a computer for $700, memory for $2000, floppy disk for $1300, ports for $500, terminals for $1500, and a cassette system for $200, totaling $6200. Since the computer is actually located at the school (our HP computer is at the district center, far from any user), teachers can provide the necessary maintenance. Hence, from a price perspective, a personal time-share system is highly desirable.

Another motivation for time-sharing arises from the observation that any microprocessor dedicated to interactive programming, such as in games, simulations, or BASIC, is mostly wasted in a one-user mode. The central processor unit is usually I/O bound; that is, it is usually waiting either for the user to type a character on the terminal or to be able to output another character to a printer or TV screen. The typical software for this wait is the "status loop" which in 8080 code often looks like:

```
AGAIN: IN   STATUS
       ANI  001
       JZ   AGAIN
       IN   DATA
```

At 500 nanoseconds per state, the CPU takes about 13.5 microseconds to check the status of a port. If one assumes a standard data transfer rate of 30 characters per second, the CPU must wait for 66,667 state times between I/O operations - during which time up to 16,667 useful instructions could be executed. Thus, instead of languishing in status loops, the computer can go off and perform a considerable amount of additional work, with no apparent loss of service to the user. In time-sharing, this additional work consists of sending the CPU off to service other users. If no user's transfer rate exceeds 30 cps, there is ample CPU time for a large number of users. It is this feature of interactive programs that makes time-sharing practical from the performance point-of-view.

There are several important advantages to a microcomputer system from the perspective of the educator. The system provides the benefit of hands-on experience for students who are interested in electronic design and construction projects. It also encourages the use of machine and assembly language programming as well as the study of computer logic and TTL and MOS electronic circuitry. The on-site computer serves as a laboratory in hardware and software for our Computer Technology course. In turn, the course supplies us with a group of students who can operate the system and take care of much of the routine maintenance, under the

Jeff Levinsky, 2240 California St. #2, Mt. View, CA. 94040.

supervision of the teacher. The availability of several terminals for use throughout the school day has already interested many students in computers, as both a hobby and as an occupation. Much of our best software has been entirely the result of student-initiated projects accomplished with little or no teacher assistance; and, as our time-share systems increase in flexibility, we expect even greater interest. Ultimately, this will create a large user community in the school, which itself easily justifies the system. And the same could well occur within a business or even a club situation.

The Concept

It is necessary to understand the concept of time-sharing on a microcomputer in order to understand how an actual time-share system is developed (experienced readers may wish to skip ahead). To start with the simplest case, imagine a microcomputer (with front panel switches) in which two simple programs reside in memory, independently of each other, and where the first program is currently running. Now, let us stop the computer and have it run the second program for a while, and then stop the computer again and attempt to resume running the first program. We will probably fail. To see why, we recall that every program uses the program counter, as well as other registers, and various sections of memory in which to store data. When the first program is stopped, the registers contain important data necessary for the continuation of the run. When the second program is executed, the values in the registers from the former program are destroyed, and so when we try to return to the first program, it finds a different set of values in the registers and perhaps in memory than those it left there. This explains the failure. The solution is simple: we must record all the values set by the first program before switching to the second, and then carefully replace these values before switching back and do the same for the second program. On an 8080, this data amounts to twelve bytes (register pairs BC, DE, HL, PSW, PC and SP) plus any memory that the programs share. The interrupt flag need not be considered. Assuming that we can obtain this information from the front panel, we can now proceed to run the first program for a while, switch to the second, and then back and forth as often and for as long as we like. We might even add other programs and rotate sequentially among them. This gives, in fact, time-share at its crudest. It now remains to make three improvements to provide a workable system.

The first improvement on the primitive time-share is to have the computer automatically save and restore all necessary data each time a transfer is made from one program to another. For this, we must define a "job": a contiguous area of memory in which all the dynamic data (that which changes) associated with a given program is stored. Thus, all of the variables that are kept in memory and the machine stack for a given program should be addressed to fit into that program's job. A typical job might look like the configuration shown in Figure 1.

Variables, buffers	Stack	Registers

Low Memory

Figure 1. Typical Job Configuration.

TIMSHR

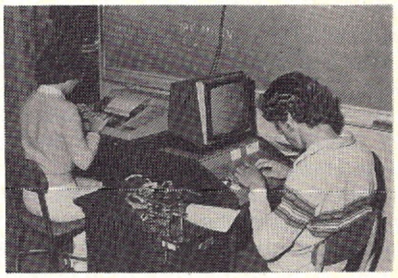
When we stop a program we now run a small subroutine called TIMSHR, which saves the values of the registers in the associated program's job. The registers can actually be saved on the stack itself on an 8080, except for the stack pointer, which must be stored at a fixed location for retrieval later.[1] The computer can now continue and service another program - first retrieving the stack pointer and the other registers from that program's job. Since each program has its own job associated with it, there is no danger now of one program interfering with another. The TIMSHR routine can, in fact, select another program from a queue after saving a program's registers and retrieve the registers of the selected program and run it, i.e., jump to it. Now, if we periodically stop the computer while it is executing a program, invoking TIMSHR will do the rest of the work.

The next improvement is a means for avoiding stopping the computer and calling TIMSHR. There are two ways to go about this. One is to use the computer's ability to be interrupted, that is, automatically forced to begin executing a pre-determined routine. A vector interrupt board may be purchased for the 8080 that provides an interrupt signal on a regular time basis, say 60 times a second. If the interrupt facility is set to TIMSHR, then periodically one program will be stopped and another chosen and run. The interrupt replaces the front panel work that we had to do before, and so the system now time-shares automatically. If the computer is primarily used to time-share interactive programs, then there is a less expensive and quite efficient way to provide the signal to switch users. As previously mentioned, the CPU spends long periods of time waiting in status loops for a ready I/O device. If a user's device is not ready for service, then the computer might as well switch to the next user. The revised status loop looks like:

```
AGAIN: IN  STATUS
       ANI  001
       CZ  TIMSHR
       JZ   AGAIN
       IN   DATA
```

If the device is not ready, the Z bit will be set, so TIMSHR will be called. This automatically saves the program counter on the stack. The last instruction in TIMSHR is usually a RET (return), so when TIMSHR has serviced all of the other programs and come back to the one containing this loop, it will continue execution at the jump-on-zero instruction and loop back to check the status again (as the Z bit is part of the PSW and is, of course, saved). If similar calls are made to TIMSHR in every status loop in each program, then the CPU will never wait for a device but will instead go to the next program in the queue and run it until that program requests a data transfer with a device that is not ready. As long as each program frequently attempts to input or output a character, each program will be serviced often. In practice, this is the case in games and BASICs and, therefore, no interrupt board is needed.[2]

Finally, we can improve upon the typical case where the programs being time-shared are almost the



Two of the CHAOS II terminals; a TI Silent 700 in the left rear and a Processor Technology SOL at the right. The strange device in the foreground is a joy stick input used by orthopedically handicapped students. We have a prototype system which is adaptable to a variety of input devices.

Time-Sharing, con't...

same. Consider several users playing the number game NIM against the computer, with each user on a separate terminal. In the system that we have developed up to this point, there is a separate NIM program in memory for each user. These programs will be identical except for the I/O port number and the location of the job associated with the program. To store nearly identical programs is very inefficient in terms of memory use and is unnecessary. To avoid this, we introduce a new condition that was not required before: the program in question must be re-entrant, that is to say, it must be free of any self-modifying code. If the program has a self-modifying code, and the time-sharing system does not explicitly handle this (it might save the modifications in the job), then one user's modifications will replace those of another user and disaster is sure to follow. As the single copy which we wish to share will address only one job, TIMSHR must now copy out the contents of the job to somewhere else in memory and copy in the job of another user before switching to that user. This is easily accomplished with a loop that interchanges each byte in one user's job with that of the other. The job currently being serviced is called the primary job and is at the fixed location addressed in the program. TIMSHR must also insure that when a certain user is being serviced, the input and output port numbers are set for that particular user's terminal. On an 8080, this can only be accomplished by storing the port number(s) in the job and by having TIMSHR poke the bytes into the appropriate IN and OUT instructions in the program when preparing to service the user. Now, each user needs only a job which results in a considerable savings in memory as the jobs are typically 1/10 to 1/20 as large as the program itself. As a final note, we replace the much over-used term "user" with the intuitive term "process." Every process is identified with its corresponding job, and we may now say that time-sharing is simply the allocation of "slices" of CPU time to each of several processes.

With all the above in mind we can now take a given program and effectively "time-share" it. Our first experiment is to time-share a simple "echo" program: one that merely displays on a terminal whatever character is typed on the terminal. The flowchart in Figure 2 illustrates the most obvious solution and does not require any jobs or calls to TIMSHR. There is a major flaw in this algorithm, however, which shows why we must require jobs and TIMSHR. Once the computer discovers that a character has been typed in, it will wait for a chance to output that character before proceeding with the next process. Should this wait be substantial, other processes may input several characters, of which all but the last will be lost. If the terminals are operating at different baud rates, the problem can become even more severe. The 8080 program listings provide a successful algorithm and incorporates jobs and a TIMSHR routine. In this instance, the job is very small (only the port numbers and registers PSW, BC, and PC need to be saved) and so registers D and SP are used to locate the primary job. Usually, the primary job contains all registers and other techniques must be used. The format of the job is shown in Figure 3.

When the program is run at START, the stack pointer will be set to the first job. From this job, the status and data port numbers will be retrieved and inserted into the proper locations in the actual echo program (STAT1; STAT2, DATA1, and DATA2).

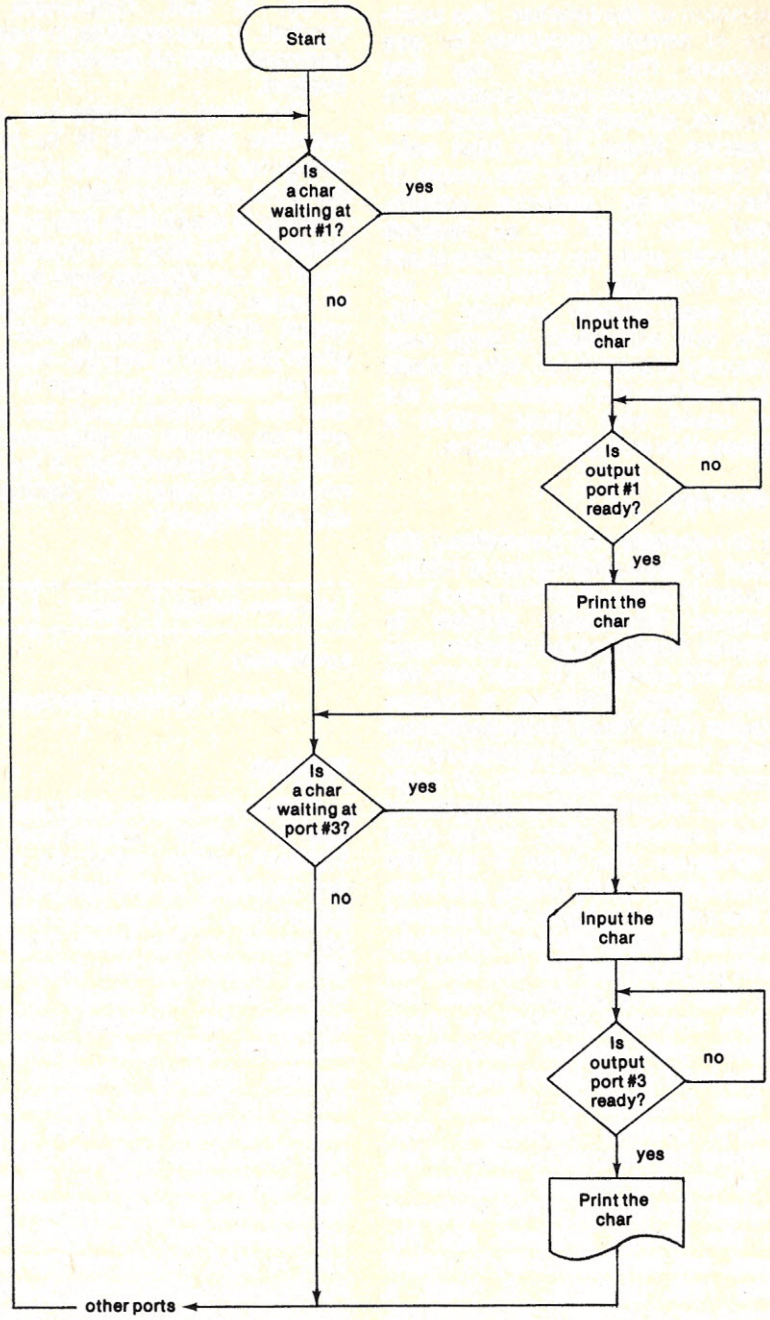


Figure 2. Echo routine flowchart.

Low Memory	Status Port #
	Data Port #
	PSW
	A
	C
Jobsize: 8 bytes	B
	PC-low byte
	PC-high byte

Figure 3. Format of Job for Time-shared Echo.

The PSW and BC register pairs will b retrieved next and the subsequent return will cause the actual echo program to begin execution. If an I/O device is not ready, a call to TIMSHR will be made so that the next job will be run. TIMSHR will save the blocked process's registers, locate the next job (by incrementing the stack pointer), and proceed as before. Register D is set originally to 1 and incremented each time TIMSHR is called. When it exceeds the number of jobs (stored in NUMJOB), the D and SP registers are reset via START. The data area must be initialized before execution with the number of jobs, and the jobs themselves. The limit of 128 jobs is due to the fact that each terminal requires two ports and that the 8080 can have only 256 ports. Barring that, and speed, there is no fundamental limit to the number of processes that can be time-shared.

Successes

After we developed an echo program we time-shared a NIM game written by Mark Doyle, one of our computer science students two summers ago. The actual time-share was written by Larry Hill, a mathematics teacher at Clairemont. In this version, both vector interrupts and status loops were used to call TIMSHR, but we later determined that the former was not essential, as I/O occurred frequently in the game itself. Since the interrupt board requires special software and the danger of interrupts occurring while the stack pointer is above meaningful stack data always exists, we now rely solely on the status loop method.

Our first major success was the time-sharing of Processor Technology's 5K BASIC in late 1976. At that time, we had four terminals and were able to operate a system using all of them. Time-sharing a BASIC requires applying exactly the same procedures outlined above. The size and location of the primary job must be determined, the status loops must be modified to call TIMSHR, and any non-reentrant code must be rewritten. Usually these tasks are straightforward, and the clear documentation of 5K BASIC made them even more so. 5K BASIC's primary job consists of a line buffer, floating point temporaries, a null count, various flags and stacks, etc. This area extends from hex addresses 1702 to 1953. The primary job must actually be a littler larger, as the registers and port numbers must be included. These are stored at hex addresses 1954 to 195E, for a total of 605 bytes. For 5K BASIC, the entire primary job looks like the layout in Figure 4. Some caution is

Hex Address	
1702	Floating Point, VDM, System, and other buffer space
1950	
1951	BOFA
1952	
1953	MEMTOP
1954	L
1955	H
1956	PSW
1957	A
1958	E
1959	D
195A	C
195B	B
195C	SP-low byte
195D	SP-high byte
195E	Status Port #

Jobsize: 605 bytes

Figure 4. Primary Job for Time-shared 5K BASIC.

needed in modifying the status loops. All BASICs allow the user to break execution of a program (in the case of 5K BASIC, by typing control-C), and thus there must exist a status check of the input port after every statement is executed. One might think that since a not-ready status indicates that the break character has not been typed and hence that execution should continue, that this status check should not be modified. However, if a program consists of few input/output operations, that program will rarely call TIMSHR and degrade the system performance. Thus, calling TIMSHR after every statement is desirable, as in no BASIC can one loop inside a statement (avoiding the status check). In 5K BASIC, there are only two status routines, and modifications are made in both of these. The VDM is not used in the time-shared version, and there is no self-modifying code to be replaced.

In BASIC, there is one additional complication: each user must have a workspace in which the user's BASIC program and variables are stored. This can be considered as part of the primary job, but then the workspace would have to be moved every time jobs are switched. 5K BASIC itself provides a more practical approach: there exist two pointers (BOFA and MEMTOP) in the primary job which designate the bottom and top of the current workspace. By initially putting different values in these pointers for each job, every user gets a separate workspace. Hence, only the pointers have to be moved, not the entire workspace. 5K BASIC allows the user to set these pointers both originally and by executing the command MEM. This cannot, of course, be allowed in a time-shared version so a jump is put around that section of code.

To run time-shared 5K BASIC, first 5K BASIC and then the time-share program below must be loaded, and the latter run at INITAL (hex address 19F4) where all of the jobs and workspaces will be initialized. First, the operator will be asked for the numbers of users wanted and the top of usable memory. The initialization code will then create the appropriate number of jobs and evenly divide up the remaining memory into workspaces, putting pointers to these in the jobs. The operator will be asked to input the status port number for each user. The port is automatically initialized, and the number stored in the job. Finally, various modifications will be made in BASIC so that the proper status routines will be used, etc., and then time-sharing will

Time-Sharing, con't...

commence. Figure 5 shows the contents of memory before initialization, a typical initialization dialogue and the configuration of the jobs and workspaces afterwards. Note that all input should be in decimal and that the first values were rejected as there would have been no room for the workspaces. To save space, the initialization program disappears when time-sharing actually begins, and so the time-share program must be reloaded whenever the number of users or memory size are to be changed. Response time of the system may be adjusted while it is time-sharing by modifying the INTLIM byte at hex address 19F3. TIMSHR will switch users only after INTLIM calls are made to TIMSHR. The default value of INTLIM is 1. If INTLIM is set to 0, the TIMSHR will stop switching users, thereby freezing in the current one, without any harm to the others. In situations where unexperienced users are on the system, the CALL instruction can be disabled (by putting a OD at 08B9) to prevent execution of random code.

MEMORY CONFIGURATION BEFORE INITIALIZATION

5K BASIC	Primary Job	TIMSHR	INITAL	

Low Memory

INITIALIZATION DIALOGUE

```
HOW MANY USERS? 0
TOP OF MEMORY? 10925
HOW MANY USERS? 3
TOP OF MEMORY? 10925
STATUS PORT? 0
STATUS PORT? 2
STATUS PORT? 4
```

MEMORY CONFIGURATION AFTER INITIALIZATION

5K BASIC	Primary Job	TIMSHR	Job	Job	Workspace	Workspace	Workspace

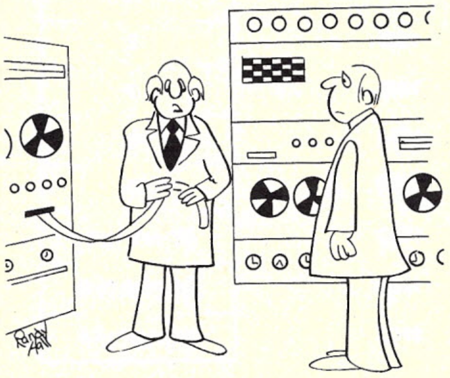
Low Memory

Figure 5. Initialization of 5K BASIC.

Conclusion

The CHATS system, written in Altair Basic by Chuck Dayton, a former Clairemont High School student, is very similar to the time-share of 5K BASIC, although the task was more difficult as MITS does not provide listings of its BASICs. CHATS uses an improved TIMSHR which checks the device status of processes waiting to perform input or output and then only sets up the process's job if the device is ready. This saves a great deal of job switching time. Since CHATS was written, we have developed techniques to switch jobs via hardware: the job number is simply sent to a parallel output port, and the job switching occurs instantaneously. As of this writing, the finishing touches are being put on CHAOS by its designer, Mary Kroening, a current student and last year's AEDS programming contest grand prize winner. The command and file facilities of CHAOS are modeled after those in Bell Lab's UNIX operating system, and like UNIX, we plan to include a variety of languages in the system, including APL, PASCAL/S, and an interactive 8080 simulator/debugger. CHAOS can handle up to eight disk drives and up to sixteen users and could easily be modified for even more. All of the writers of our systems have agreed to make tapes, listings, and documentation available: inquiries should be addressed to the Clairemont High School Computer Club, c/o Robert Hass, 4150 Ute, San Diego, CA 92117. Please note that we cannot guarantee that our systems will operate on equipment we are unfamiliar with and that we cannot provide any information on the BASICs themselves.

We believe our efforts indicate that the simple time-sharing presented here opens up many powerful and economical ways to use microcomputers for experimenters, schools, clubs and small businesses. Word processing, accounting and instructional programs can be as easily and efficiently modified for several users as BASICs. No special hardware is required and the only cost of adding a user is for the terminal and the I/O port. Currently, a few manufacturers are advertising dial-up ports for the S100 bus, and these ports could be combined with time-sharing to provide clubs with around-the-clock facilities for their members and to provide businesses with a master inventory and accounting system for remote stores. Space and computer time on a disk time-sharing system could be sold at a substantial profit. As faster microcomputers and less expensive mass memories become available, there is no reason why micro-time-sharing cannot grow to rival large scale systems in performance and usability while remaining far below them in cost. □



"It says that due to the demeaning nature of the task, it refuses to start your morning coffee." ©Creative Computing

Footnotes

1. Storing the registers on the stack can lead to stack overflows and can also destroy stack data if the SP is used in certain ways.

2. Interrupts can, however, provide efficient I/O buffering that can make read-ahead possible, for instance.

TBS DEALERS HAVE FULL SYSTEMS SUPPORT.

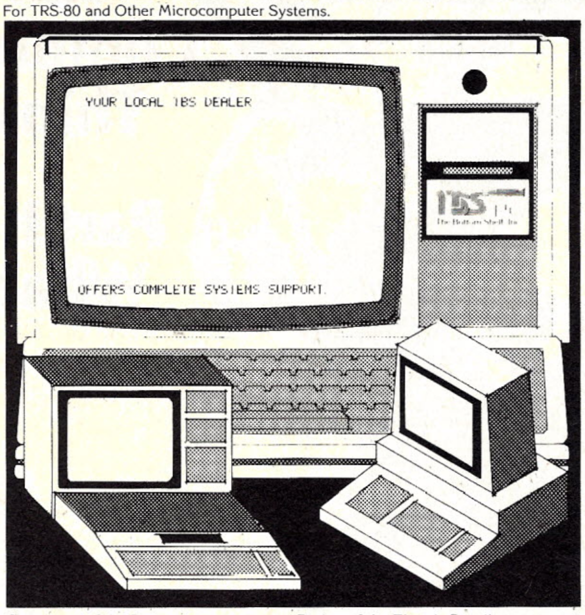
You've just purchased a Centronics printer. Where do you get the printer supplies needed to keep it functional? You've just purchased a disk drive. Where do you find a source of quality diskettes and storage devices to keep them organized? You have invested over $1,000 in your system. Where will you find all the necessary items to keep it operational? The answer: TBS and its affiliate DEALERS.

The LIBRARY 100 Software Pack was published in November of 1978. Since that time, it has been widely accepted throughout the United States and 31 countries as a major advancement in microcomputer software. Now, we at TBS have expanded our endeavor to better meet the product needs of our customers.

SYSTEMS EXTENSIONS

$3.00

For TRS-80 and Other Microcomputer Systems.



Computers of the Past
Computers of the Present
Computers of the Future
Methods to Program Your System
Software Background Report
Preparation for Programming
Standards for Professional Programming
Security of Business Data and Programming
Purchase, Care and Maintenance of the Business Computer
Review of the Electric Pencil
The Diskette Revolution
The TRS-80 and the Business Community
Computer Aided Instruction
The Microcomputer and the Wall Socket
Level II Index
Random Ramblings

A camera is useless without film; a typewriter is useless without paper; a computer system is useless without the necessary support to make it completely operational. TBS is leading the way in systems support, and we are proud to announce a new concept for the microcomputer industry - FULL SYSTEMS SUPPORT. Mainframe and minicomputer industries have been following this concept for years. However, until now it has not been available for microcomputer systems. TBS Full Systems Support is aimed at helping you, the computer owner, acquire the necessary support items quickly and inexpensively through a network of affiliate dealers nationwide.

We also realize that there is limited material available on computers in general. To fill these needs we have produced SYSTEMS EXTENSIONS. This is a unique publication which provides a theoretical computer background for the novice computer user and some specific points of programming interest for TRS-80 users. Seventeen articles are included in 63 pages of solid text.

But even more important, SYSTEMS EXTENSIONS provides a source of supply for a series of products applicable to all computers. Through this publication TBS and its affiliate dealers offer over 300 items including software, diskettes, cassettes, printer supplies, storage units and furniture. In other words, TBS and its dealers stock everything to make the microcomputer a full scale business system.

At the heart of your computer system is the software. We now offer the following programs for the TRS-80: **TBS Business Mail System** for dual disk and printer; **Check Register Accounting System** for dual disk and printer; **Basic Toolkit**, a programmers aid; **System Doctor**, a computer diagnostic program; **Checkbook II**, cassette based personal finance; **Information System**, an "in-mem" data base manager; **Analysis Pad**; a columnar calculator; **Terminal Control**, for RS-232 tele-communications; **Exerciser**, for establishing physical fitness goals and regimens; plus, **Electric Pencil**; **Fortran** by Microsoft; **NEWDOS+**; and of course, the **Library 100**. Details in full are included in **SYSTEMS EXTENSIONS**.

We at TBS believe in the microcomputer revolution. Our purpose is to serve you, the microcomputer owner and user, by supplying the necessary items you need. To this end we have published **SYSTEMS EXTENSIONS**, and it is available nationwide through TBS dealers and Radio Shack Associate stores for only $3.00. For further information please call us at the number below.

TBS INC.

The Bottom Shelf, Inc.
P.O. Box 49104
Atlanta, Georgia 30359
Phone: 404-939-6031

CIRCLE 114 ON READER SERVICE CARD

The Controversy Goes On — "Can Computers Think?" Part II: What Is Thinking?

Peter Kugel

People had wondered whether machines could think long before the computer was invented. They wondered about it for at least two reasons. One was practical. They wanted to know whether it might be worthwhile to try to build machines to do some of our thinking for us. The other reason was theoretical. They wanted to know whether the human mind might be a machine, because if it was, then human thinking might be explainable in merely mechanical terms.

Those who pondered the practical question were divided over whether thinking machines would be a good thing to build. On the one hand, it seemed that it would be good to have machines do some of our more routine thinking for us. On the other hand, such machines might leave us with nothing to do. They might even take over the world.

Those who pondered the theoretical question were also divided. If the mind could be described in mechanical terms, then such descriptions might lead to a better understanding of the human mind. But others were not so pleased by this prospect because they felt a reduction of the mind to "just another machine" would demean us even more than Copernicus' reduction of the Earth to "just another planet" or Darwin's reduction of mankind to "just another animal species" had done.

Those who understood the work done by the British mathematician Alan Turing during the middle of the 1930's were even more disturbed by the insulting implications that such a reduction would have. Turing had shown that some strikingly simple machines, that we now call "Universal Turing Machines," could think if any kind of machine could think. How simple these machines really are is suggested by the fact that when Joseph Weizenbaum of M.I.T. described how they worked in his book **Computer Power and Human Reason** he did it in terms of a machine constructed merely from a roll of toilet paper and a pile of pebbles. A reduction of the human mind to such a simple machine seemed like some kind of ultimate insult.

The utter simplicity of the Turing machines may have bothered those who thought about the question "Can machines think?" in theoretical terms, but it delighted those who thought about it in practical terms. It meant that such machines would be easy to build. The importance of this aspect of the matter was soon to become apparent. There was a war brewing in Europe and this war was to lead to some inventions that were to change the texture of the "Can machines think?" controversy by turning it into a practical matter.

In September of 1939 Germany invaded Poland and World War II began. With a war on, people began to

think up uses for the machines that Turing had only imagined. In Nazi Germany, Konrad Zuse thought that they might be useful for doing the aerodynamic calculations that were to lead to the first jet airplanes and space rockets. In America, John von Neumann and others thought that they might be useful for doing the computations that led to the atom bomb. And in England, Alan Turing and others went to work, building computers to break the German High Command's secret codes. (The story of this is told in F.R. Winterbotham's book **The Ultra Secret**).

After the war, when computers were a reality, Turing began to think about the question "Can computers think?" In 1950 he wrote the famous paper in which he tried to define that question as, some fifteen years before that, he had defined the computer. "What," asked Turing, "would we ask a computer to do before we would say that it could think?" Turing suggested that a computer could be counted as thinking if it could duplicate what we would today call a human being's "information processing behavior." Turing defined this neatly. A human's information processing behavior is what he or she can transmit over a teletypewriter terminal. A computer could be said to think if it could do the same things via a teletypewriter terminal that a person can and, as a test for this capability, he defined what we now call the "Turing Test." We will count a computer as thinking if it can pass this test which goes like this. We put person A into one room, computer B into another, and close both doors. Then we bring in a human judge and allow him or her to communicate with A and B via teletypewriters only. The judge does not know which of the two rooms contains the computer and the judge's job is to try to use information obtained through the terminals to determine which is which. Of course, the computer is allowed to lie. The judge can ask the occupant of either room "Are you a computer?" and presumably both will type back "No." Or the judge can ask both to write a poem, name their favorite ball team, prove a theorem or what have you. If the judge cannot tell, after some given period of time, which of the two rooms contains the computer, then the computer passes the test and is counted as thinking. The purpose of the teletypewriter is to allow only the information processing behavior of the two systems to be compared. No sound of voice or color of hair is allowed to count.

Can a computer program pass this test? None has really done it yet, although some have come surprisingly close. But many workers in Artificial Intelligence defined their research problems on the basis of this idea, and much work in Artificial Intelligence can be thought of as attempts to develop computer programs for passing Turing's test, one part at a time.

There are several reasons for wanting to try to program a computer to pass Turing's test. One is that, in the process of trying, one learns a good deal about thinking. What we have learned so far has added to our understanding of thinking and promises to add more as the years go by. Another reason for trying to program a computer to pass Turing's test is that as one develops machines to pass various parts of the test, one learns more about how to get computers to do intelligent things. Although there haven't been all that many practical applications of the things we have learned in this way, the kinds of performance we can now get out of computers is impressive when it is compared to what we were able to get only a few years ago.

There are also some reasons for **not** wanting to try to program a computer to pass Turing's test. One is that it might lead to intelligent machines and one might not view this possibility as particularly desirable. Another reason is based on exactly the opposite premise — it might turn out to be impossible to develop an intelligent machine and if this is the case, then one will have wasted a lot of time. True, if people were to try to develop, for example, a thinking potato, one might question their wisdom. A potato has so little information processing power, that one would be tempted to warn them that they were wasting their time. But the idea of a thinking computer is quite plausible and, barring more convincing evidence that it too lacks enough information processing power, it seems to make more sense to assume that a computer can be programmed to think than that it can't. So some computer scientists began to work on this rather optimistic assumption.

While computer scientists in a few laboratories were developing programs to produce thinking behavior, (i.e., to play chess, write poems, understand English, answer questions from IQ tests and do some of the other things that computers might be called on to do to pass Turing's test) most computer scientists ignored the question that Turing had intentionally ignored: "What is the nature of intelligent thinking?"

It is clear that Turing's Test does not serve to define such thinking. A machine can pass Turing's Test without being able to think on the one hand and a machine that can think could fail to pass Turing's Test on the other. A machine could pass the test if the judge were stupid enough, if it had a way to read the judge's mind, or if its programmer had a way to predict the specific questions the judge was going to ask. Or a device could be intelligent and fail to pass the test if it only spoke Turkish while the judge only understood English. An intelligent being from outer space would almost certainly fail. So the ability to pass Turing's Test is neither a sufficient nor a necessary condition for being able to think and Turing never thought that it was.

This bothered some of the philosophers around at the time for it is the business of philosophers to be bothered by questions that other, more practical, people ignore. As these philosophers were aware, there has been a long tension in the history of Western thought between those who argued that man was some kind of machine and those who argued that man was "something more." The urge to reduce the workings of man's mind to the workings of a machine is essentially the urge to place man's thinking into some sort of framework that is susceptible to scientific investigation and the hope of doing this is what has led people to try to identify the mind with the most impressive machine of their day.

Most of these attempts have been useful in one way or another but, as those who argue that man is something more than a machine keep noting, there always seems to be something left out of the machines.

This time, however, the situation seems different. The argument that there are more powerful kinds of machines no longer seems available to us, thanks to Turing's Thesis. With the lines drawn this sharply, most reasonable people accepted the rather hopeful view (hopeful because it puts the human mind within the scope of scientific investigation) that computers could think. And people at M.I.T., Stanford, Carnegie-Mellon, Edinburgh, and a few other places, settled down to program them to think, a bit (no pun intended) at a time.

Before I go on with the story, stick a little thought in the back of your mind. Notice how the claim that computers can think, rests on the claim that either computers can think or no machine of any kind can think and notice how this, in turn, depends on Turing's thesis that says that there are no machines that can do things that Turing machines (or computers) can-

Controversy, con't...

not do. If this thesis is false, and it has withstood some 40 years of searching criticism, the issue now may be no different than it was the last time some theory of the mind as machine failed. All that would be necessary would be to turn to some more powerful kind of machines as the basis for one's mechanical theory of the mind.

The initial successes in programming computers to think were quite striking. Programs that could play chess and checkers were demonstrated. So were programs that could solve a large variety of problems, prove lots of theorems, translate from Russian into English, answer questions about various subjects, and so forth. Things were beginning to look pretty good for those who thought that the answer to the question "Can computers think?" was "Yes."

The general feeling of optimism was quite intoxicating and people began to get intoxicated. Some expressed optimistic hopes too strongly. Others lied, or at least shaded the truth. Robots to take over all sorts of jobs were promised soon. And the popular press was beginning to pick up, and accept, some of these extravagant claims.

This was the environment in which Hubert Dreyfus, one of the first really vociferous critics of Artificial Intelligence, found himself when, in 1965 at RAND, he began to look at the status of Artificial Intelligence from an outsider's point of view. Dreyfus taught philosophy at M.I.T., one of the centers of Artificial Intelligence research. Being an existentialist and a phenomenologist, he was used to trying to study experience as it was experienced and trying not to break it down into simpler components. So he told the Artificial Intelligentsia that their components weren't adequate. Simple machines that understand only 0's and 1's are inadequate to duplicate human experience, he claimed. So are machines whose registers register only one thing at a time. People are aware of many things at once and without this ability, computers cannot think. And look at all the richness of human experience that computers are missing. They can't even walk around in a room. So how can they understand the world like a person?

And perhaps most tellingly, he showed how extravagant some of the claims that had been made in favor of thinking machines had been and how mediocre the achievements were in comparison. The computer does not think today, he pointed out (quite correctly), and it cannot be made to think tomorrow because it lacks some of the basic capabilities that thinking requires. The machinery of the Turing machine, according to Dreyfus, is not enough.

Early successes in producing programs that could do part of the job of thinking, argued Dreyfus, together with the underlying belief that the only difference between what computers do and what we do is that what we do is more complex (but still of the same kind), had led some people to think that any human thinking activity could be expressed as a computer program. But problems have arisen and these are not problems due to the laziness or ignorance of the workers in the field. The trouble lies, Dreyfus suggested, in the fact that the "vocabulary" of the computer is not adequate to express certain kinds of things that people can do and know. The things that can be done by computers thus leave out some things that the mind can do because some of the basic machinery required seems to be missing. Go back to the brain, argued Dreyfus, and try to build computers that have some of the special machinery of the brain.

In short, he argued, those of you who are trying to program intelligent computers are trying to do it with inadequate tools. Although Dreyfus did not use this example, the situation is somewhat like that of the average person trying to solve the problem of trying to draw 4 connected lines, without lifting the pencil from the paper, so that they pass through all the dots in this figure:

. . .

. . .

. . .

As long as one assumes (as most people do who have not seen this puzzle before) that one has to "stay within the square," one cannot succeed. It is only when one stops limiting oneself to staying within the boundaries of the square that one can solve this problem.

In this problem, one sets one's own inner restrictions that the problem does not. Dreyfus is arguing that, in the case of Artificial Intelligence, one has explicit limitations that make a solution impossible. Try to program a thinking computer without using (say) analog procedures and you must fail. Why? Because the analog procedures allow you to do things that the digital ones do not. The solution, according to Dreyfus, is to extend one's idea of what a machine is beyond the digital computer. Unlike the people in Artificial Intelligence, who try to do the best they can with the existing machines, Dreyfus was opting for new ones.

Dreyfus may have thought of himself as voting for more powerful tools but that is not the way he looked to much of the Artificial Intelligence community. For, if one accepts Turing's Thesis, to say that computers cannot think is to attack the scientific method itself and to put thinking beyond the reach of reason. Look at what Dreyfus is saying through the eyes of a worker in Artificial Intelligence. If he is saying that you need some kind of machinery other than that of the Turing machines or the digital computer, then he is being stupid since Turing's Thesis says there is no such machinery. He was criticized on this ground. If, on the other hand, he is saying that no machinery can be built to think because thinking is not reducible to any kind of mechanical thing, then he is arguing that thinking falls outside the scope of reason and therefore leaves thinking to the non-scientific mystics. And he was criticized on this ground too.

Dreyfus was neither the first critic of the view that computers could be programmed to think nor was he the last. One important group of critics were the authors of the Lighthill Report who argued that putting money behind the work in Artificial Intelligence was a waste because it wasn't going to pay off in anything useful. Another was Joseph Weizenbaum in his recent book **Computer Power and Human Reason**.

Dreyfus' criticism of the work in Artificial Intelligence is from an outsider. Weizenbaum's is not. Weizenbaum is (or at least was, before he published his book) one of the favorite stars of the Artificial Intelligence community and he knows a good bit about computers and programming. One of the computer programs Weizenbaum wrote came just about as close as any program yet has to passing Turing's test.

Weizenbaum's program is one of the most widely demonstrated programs there is. It acts like a psychiatrist, talking to you (via teletypewriter) about your problems. A conversation with it might go like this: "Do you have any problems?" it asks. (This is not a transcript of a real "conversation" with it, but it conveys the general idea.) "Yes" you type "I don't like my mother." "Tell me about your mother," says the machine. And so it goes in that general vein.

The program is really based on an old trick that people use when trapped by boring guests at cocktail parties.

You don't have to listen to a conversation to seem to be participating in it. You just have to repeat back the other person's words or keep saying "Hmmm."

There is the famous, though possibly apocryphal, story about Marvin Minsky's secretary at M.I.T. to whom Minsky (one of the best known workers in the field of Artificial Intelligence) was demonstrating Weizenbaum's program. After a few minutes she asked him to leave the room because the conversation was starting to get personal.

That's all there is to the story and what is so funny about it to some people is that Minsky's secretary (bless her) took this mechanical program as seriously as a human being. That is the kind of reaction that helped prod Weizenbaum into writing his book. Minsky's secretary is not unique in her ability to be fooled by this program. Gardner Quarton (and others), then at the Massachusetts General Hospital, let a number of people "talk" to Weizenbaum's program for a while and then asked them to say whether they had been talking with a person or a machine. A majority said they thought they had been talking with a person.

Here is a version of Turing's Test that a machine passed. This was largely ignored as a piece of evidence for the claim that computers can think because there is a sense in which it points out one of the flaws in Turing's test: the judges can be misled. The experiment showed how easily this can happen. And, of course, the result does not constitute a successful attempt to pass the Turing Test on at least two counts — the subjects were not told about the fact that they were participating in a test and they were not given a human for purposes of comparison.

Weizenbaum's critique of Artificial Intelligence is different from Dreyfus's for at least two reasons. One is that it comes from somebody inside the computer field, which makes it a bit more "biting." The second is that, where Dreyfus is attacking the basic Turing machine model and suggesting that things be added to it, Weizenbaum is warning us to look at how that model is used. Weizenbaum is warning us not to take the "man is only a machine model" too seriously because any model is just that, a model that leaves out something in the original. Weizenbaum's criticism would apply to AI even if it accepted Dreyfus' criticism and used more powerful machinery.

Weizenbaum's criticism is a criticism of a large variety of models of thinking. It covers other machine models too. People who are not scientists often overlook the central role of the model in scientific thought. When a scientist studies a phenomenon on X (say thinking) he or she develops a simplified model of Y of X. That model Y might be a description in terms of ordinary English, or it might be described in terms of a set of mathematical equations, or in terms of a computer program. The scientist then focuses his or her attention on the model and studies that. Y is not X, of course, but it resembles it and for many purposes that is enough.

Models are easier to study than the real things since they are both simpler to construct and simpler to manipulate. That is why there are wind tunnels to test model airplanes and why people make scale drawings of their living rooms to see where things can fit before moving the real (and often heavy) furniture in their living rooms around.

Models always differ from the real thing they model because they leave things out. But, even though most scientists know this perfectly well, there is something tempting about forgetting it and taking scientific models too seriously.

We all do it from time to time. When one discovers a new scientific model, it is very easy to try to see everything in terms of the model one has just found. The idea of computation is no exception. When one first learns about computer programs, one commonly begins to see lots of things in terms of such programs.

To have a good model of something in the world gives you a feeling of mastery over that thing. You feel you understand it. You have to work with a model for quite a while before you notice the things that it does not explain. Indeed you may never notice them. For example, you probably think that you have an adequate model for what causes tides. It is (you think) the gravitational effect of the moon, pulling the oceans. Fine. But there are two problems. One is that there are two tides a day and only one moon. (So what pulls the water the second time?) The other problem is that the gravitational pull of the moon is not strong enough to pull oceans up even as much as an inch and tides are much higher than that.

We all accept models of things that do not fully fit them although some of us do it with more enthusiasm than others. Not that such enthusiasm is necessarily bad. Any scientist worth his or her salt gets enthusiastic about the models of his or her science. Teachers of science try, rightly, to encourage such enthusiasm in their students. What Weizenbaum is criticizing is what happens when this enthusiasm goes too far. It goes too far when you think that all there is in the real world is what your intentionally simplistic model tells you about.

There are good reasons why our models of the world are simpler than the reality. They have to be if they are to serve their purpose. Our models of things are what we manipulate to understand the things that are being modeled. If they weren't simpler than the original, we wouldn't need them and probably wouldn't use them.

A great deal of what we know about the world and what we learn in school can be described in terms of models. We know a lot about matter because, thanks to the Greeks, we have an atomic model of matter which allows us to describe matter, in its various manifestations, in terms of what used to be 92 different kinds of atoms. We know a lot about human genetics because we have a good model of the basic molecules involved in it.

It has been a lot harder to find good models for human thinking than for physical things. Thinking is much harder to see or point to or disect. So, since human thinking resembles what goes on in various kinds of machines and, since we can open up machines and understand them (which we cannot do with the human head), it makes sense to try to build a conceptual model of the mind in terms of whatever machines were around for that purpose.

There are several reasons for using machines as a model of the mind and it is important to recognize that, although none of the machines that we have used as models for the mind before the computer may have been totally successful, the use of even the partially successful machine models has contributed to our understanding of the mind.

One of the reasons that machine models help us is that machines have parts that we can point to and see, parts that we can describe precisely and methods of operation that follow certain rules. In contrast, human thinking takes place in the head whose parts are not easy to point to or describe precisely and whose method of operation is largely unknown. By saying that the mind is like such and such a machine, we gain considerably in our ability to talk about thinking clearly because we can use machine terms to talk.

Another reason for liking machine models is that we can test theories of thinking by asking the person who

Controversy, con't...

puts a given theory forth to describe how he or she would build a machine to implement that theory. If the description is forthcoming and it seems to work, we can then consider the theory further. If not, we can dismiss the theory as too vague or wrong.

Machines can be built and because of this, they can serve as referees between competing theories of thinking. You suggest that thinking is done in such and such a way? Okay, you build a machine (or describe it) that works in that way and then, if the machine you describe produces intelligent behavior, that supports your claim. (It doesn't follow from such a demonstration that the mind does work in the way your machine does, but it follows that it could.) If your machine doesn't think, then that suggests that your theory leaves something out.

These reasons that make machine models of the mind useful support the idea of using the computer as a model for human thinking. But those who claim that computers can think make a further claim. This claim is that finally, here in the computer, we at last have a machine that, unlike the telephone switchboard model used by American behaviorism or the hydraulic (Viennese sewer) model used by psychoanalysts, will finally be adequate to serve as a model for all of human thinking.

Turing's Thesis says that the computer can do anything that can be done by any machine at whatever (in the way of information processing) supports this claim. And so do some of the striking differences between the computer and all the machines that have gone before it.

Most, if not all, of the machines before the computer were proposed as models of the mind tended to be rather limited and inflexible. It is this pre-computer machine that most script writers have in mind when they portray the clunky and clumsy robots we see in movies or on television. But the computer has several features that seem to surmount these shortcomings of other machines.

Most of these shortcomings are encapsulated in the layman's objection that "Computers cannot think because they can only do what they are programmed to do." Anybody who has written computer programs has several reasons for feeling that the computer really is capable of transcending the programs we write for it.

One source of the computer's apparent ability to transcend the programs we write for it, is due to its enormous complexity. It took a while for some people to realize how complex the computer could be even though its basic building blocks were so simple that they could be described in terms of a roll of toilet paper and a pile of pebbles. The complexity comes from the fact that these simple elements can be combined in such enormously varied ways. One way this fact manifests itself is that we can write a program for the computer, look it over carefully, and still not know how that program will make the computer behave. Even if a computer only does what we tell it to, then it remains quite unpredictable indeed, for even the person who tells it what to do, cannot say what the implications of that telling are. This is one reason why people who program computers tend to take the complaint that "Computers only do what they are told to do" quite lightly.

Another source of the computer's flexibility is the conditional branch which allows a program to take different actions depending on how the results of tests it makes come out. This means that the actions of the program need not be determined by the program alone, but can also be determined by the environment in which the program is run.

To see what this means, consider the behavior of a piece of clockwork machinery built to negotiate a maze. We know that if we can make it big enough, we can construct such a piece of machinery to negotiate any maze we might choose to build. But we also know that we can build a computer-driven robot with a rather small program that can, by repeatedly looking for openings in a systematic manner, find its way through any maze drawn from a very large class of possible mazes. It is the conditional branch that makes this possible (and this is why a computer can do such things as control an oil refinery).

Another source of a computer's flexibility arises from its ability to lay down memory traces and to use its conditional branches to change its behavior as a function of what it has "learned" from its past. To see what this means, consider Samuel's checker playing program that began its "life" as a mediocre checker player. As it played more and more games, it changed a table that controlled its play. After practicing against itself during the evenings (when, in the early days, computers were not used for commercial purposes) it eventually managed to learn to beat Samuels (and most other opponents) regularly.

Not only can computers change the values that their programs refer to, like Samuel's program, they can generate their own programs and then run them. One of the really significant features of the digital computer is that it can treat programs as data. This is what makes possible the large number of higher level languages like Fortran, Cobol and Basic. It also promises to make possible computers that program themselves. And when we have those, what happens to the argument that "Computers are only smart because their programmers are"?

Still another way that a computer can incorporate considerable flexibility into its behavior is through the use of a random number generator which allows one and the same program to produce different behavior each time it is run even without any changes in its input.

These last two features of the computer are often used to support the view that computers can become self-conscious or exhibit free will. Computers can be thought to become self-conscious when they operate on their own programs (although one might doubt that it is all that simple) and they can be thought to demonstrate freedom from deterministic rules when they use random numbers (although neither Turing nor I consider this a viable argument for free will in computers).

All these features of the computer make it seem so much more flexible, adaptable and even possibly more human than the machines that preceded it. But are they really all that is required for thinking? Might there not still be more powerful machines than computers that come closer to being like us in their information processing capabilities?

I want to suggest that, impressive as computers may be, they are not the ultimate information processing machine and that the core of the remaining "Can computers think?" controversy lies in this. To see why this is so, we need to look, for a moment, at what people do when they behave intelligently.

The first thing to notice is that although people aren't programmed, they seem to be directed by something that behaves much like a program. Thus, for example, when people learn to play a game like chess, what they have learned to do is almost certainly represented by some sort of finite configuration in their heads that they can apply to a particular position in chess to generate a next move. This finite configuration is not in general, a list of all possible situations and what to do in each one.

Rather, it determines a procedure that can be used to look at a chess position and generate a "best" next move for what is, in practice if not in fact, an infinity of possible positions. It is, in other words, something very much like a program.

One of the striking successes of the work in Artificial Intelligence has been the construction of computer programs that duplicate this ability fairly well. Recent successes of these programs suggest that they are moving well up to championship levels. If the ability to play good chess is evidence of intelligence, these computer programs might seem to come out as more intelligent (in this narrow area) than the people who they can beat.

I want to try to argue that they are not really intelligent. A chess playing computer that beats me, and does a lot of other things by way of information processing better than I do (I already know it does arithmetic better than I do and I think when I do arithmetic) may still be less intelligent than I am. The reason is that I had to develop my own programs (or brain configurations) to do these things, while the computer had them given to it.

For a simple example of what is involved here, consider the person who solved a puzzle quickly as against one who solves the same puzzle more slowly. Even if that puzzle is one whose solution requires intelligence, the faster solver may not be more intelligent. Suppose, for example, that somebody had told the faster puzzle solver the solution ahead of time.

I am going to suggest that intelligence is the ability to develop general ideas (or the programs that represent them) and not the ability to apply those programs (although obviously the ability to apply the programs is the only thing that makes the deriving of them worthwhile).

I do not deny, for one moment, that the computer exceeds us in its ability to apply general ideas once they have been found. But does it come up to our ability to derive them? I want to show why I think it does not and why that fact, in some sense, takes it out of the "intelligence sweepstakes."

Let's start out by discussing why it seems that the ability to come up with ideas is at the heart of what we consider intelligence and then we can turn to a discussion of why it is that computers can't do it.

It is quite common to take, as a sign of intelligence, the ability to perform "difficult" information processing tasks - prove difficult theorems, play excellent chess, understand physics and the like. It is therefore common to take intelligence to be "merely" the ability to handle "difficult" problems and I think that this view underlies much of the current work in Artificial Intelligence. However, the following cases seem to suggest that this idea, however appealing it might seem at first, is wrong:

I. The student of physics today, who knows more physics than Newton ever knew, is not thereby demonstrating that he is more intelligent than Newton was even if he or she "knows" all the physics that Newton did (and more).

II. A computer program that prints out a proof of (say) Goedel's incompleteness theorem, or the student who copies it, are not thereby considered to have demonstrated their intelligence, although Goedel, when he wrote out the same words, was.

In these cases, the behavior is the same whether or not it demonstrates intelligence. What seems to differ is the inputs (or givens) from which the behavior (or the capacity that the behavior evidences) stem. I and II suggest that the ability to do difficult things is not necessarily a sign of intelligence if it was achieved in a relatively simple way.

III. There are feeble-minded people, called "idiot savants," who can perform certain quite difficult information processing feats (usually computations) but are still considered to be feeble-minded.

Nor it is necessary that the abilities that demonstrate intelligence be very difficult. They can be quite simple.

IV. Although the ability to distinguish trucks from aeroplanes is not usually considered to require great intelligence, the sufficiently young child who suddenly acquires (without explicit instructions) the ability to distinguish trucks from planes, and to call each by its right name, is often thought to demonstrate its intelligence by this feat (and this cannot simply be put down to parental pride). Note that this appearance is diminished if the child is trained to make this response, if the child is older or if the differences have been explained to it.

V. The first-grade student who discovers that multiplication is symmetric (a x b = b x a) by merely looking at the multiplication tables, demonstrates his or her intelligence even though he or she cannot express this "result" very clearly. On the other hand, the graduate student who "discovers" this fact after a year of abstract algebra and, furthermore, is able to say it much more clearly than the first grader, might be evidencing brain damage rather than intelligence, for having taken so long.

The behaviors (if we ignore the histories leading up to them) are similar but what they say about the intelligence of the people displaying them is quite different. Again, what appears to be different is how the results were obtained. The intelligence that would have been demonstrated even by the first grader who learned the symmetry of multiplication by being told about it, would be considerably less than what he or she would have demonstrated by figuring it out.

Although what I am saying might be summarized as a proposal to identify intelligence with the ability to "learn," it is important to distinguish the kind of learning that I have in mind from what is often referred to by this word. I am not trying to say that intelligent machines are machines that somehow learn only after a series of failures if it is possible (with the information given) to learn what is required without these failures. Nor am I saying that machines that "forget," or have other difficulties, are therefore intelligent. The kind of learning that I want to be identifying with intelligence is the learning of concepts from examples. Here errors may be necessary because the examples do not (and, if the concept covers in an infinite number of cases, cannot) completely specify the concept. This is a basic feature of certain kinds of problems on intelligence tests. For example, one may be asked, on such a test, to continue the sequence 2,4,6,8... and one is expected to say: "10,12,14,...". But there are many (indeed infinitely many) perfectly good continuations. What such a test asks us to do is to find the "correct" pattern that underlies the given sample. It is the ability to do this kind of thing that I propose to identify with intelligence. It is important to notice that there is no absolutely "correct" pattern, although we tend to think that some patterns are "better" than others.

I propose to identify intelligence with the ability to **figure out** a procedure rather than with the ability to **carry out** a procedure after being told how to do it. Of course, this proposal remains vague until we can distinguish, in a precise way, the difference between "figuring out a procedure" and "being told how to carry out a procedure." □

JADE Computer Products

FREE! A 12 INCH, BLACK AND WHITE, MONITOR WITH THE PURCHASE OF EVERY SORCERER!

The Exidy SORCERER

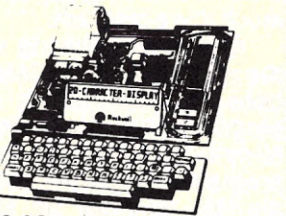
Flexibilty is the key. The Sorcerer Computer gives you the flexibility of using ready-to-run, pre-packaged programs or doing your own thing and personalizing the programs for yourself. Which ever you choose, the Sorcerer is the personal computer that speaks your language.

The Sorcerer also provides full graphics capabilties. Each character, formed by an 8 x 8 dot cell, can be programmed as a graphic symbol set. High resolution (512 x 240 addressable points) gives a total of 122,880 locations for super animation and extremely tight plotting curves. The alphanumeric set gives 64 x 30 characters on the video screen

PART NUMBER: SYO-5016A

With 16K of memory $1150.00

Rockwell AIM-65: The Head-Start in Microcomputers



A KIM-1 compatible machine with on-board printer and a real keyboard!

$375.00 w/1K RAM
$450.00 w/4K RAM

4K assembler/editor in ROM: $80
8K BASIC in ROM: $100.00
Power Supply: $59.95
Case of AIM-65 $49.95

Special Package Price: $599.00
AIM-65 (4K), Power Supply, Case, and 8K BASIC ROM

NOVATION CAT ★ ACOUSTIC MODEM

$189.00

Features Include: 300 Baud Answer/Originate, Bell 103, Comes Assembled and Tested

PROTO BOARD

Includes gold plated fingers, S-100 size, holds 72-16 pin dips, accomodates all 8 thru 40 pin dip packages.
Reg -$19 95 Special Price-$16.95 PART TSX-140B

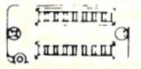
SD SYSTEMS SBC-100

An S-100 single board computer Z-80 CPU with 1024 bytes of RAM 8 to 32K bytes of PROM Serial I O port
Kit $239.95
Assembled $369.95

SD SYSTEMS Z-80 STARTER KIT

Based on the powerful Z80 CPU, this kit is an ideal introduction to microprocessors. It has an on-board keyboard and display, plus cassette tape interface and expansion provisions for two S-100 connectors. This "Do-it-all" Board will also program the 2716 2K EPROM.
Kit $249.95
Assmbld and Tstd $399.95

TEXTOOL ZIP DIP® II Sockets



PRICES
16 pin Zip Dip II $5.50
24 pin Zip Dip II $7.50
40 pin Zip Dip II $10.25

ZERO INSERTION FORCE

LEEDEX MONITOR

- 12" Black and White
- 12MHZ Bandwidth
- Handsome Plastic Case

$139.00

TRS-80·APPLE·SORCERER·TRS-80·APPL

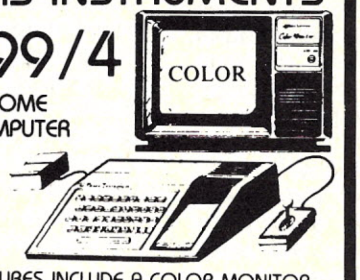
JADE MEMORY EXPANSION KITS For TRS—80, Apple, & Exidy

4116's

Everything a person needs to add 16K of memory. Chips come neatly packaged with easy to follow directions. In minutes your machine is ready for games and more advanced software.

$82.00

TEXAS INSTRUMENTS TI-99/4 HOME COMPUTER



HIGH QUALITY 13 INCH COLOR MONITOR!— Specially matched for use with the TI-99/4 console. Uses a simple, sure hook-up.

UP TO 72K TOTAL MEMORY CAPACTIY— 16K RAM, plus up to 26K ROM onboard, plus up to 30K ROM in TI's Solid State Software Command Modules.

16-COLOR GRAPHICS CAPABILITY- Easy to access high resolution graphics have special features that let you define your own characters, create animated displays, charts, graphs, etc.

MUSIC AND SOUND EFFECTS— Provides outstanding audio capability. Build three-note chords and adjust frequency, duration and volume quickly and simply. You can build notes with short, straightforward commands. Five full octaves from 110 Hz to beyond 40,000 Hz.

BUILT-IN EQUATION CALCULATOR— Unique convenience feature helps you find quick solutions to everyday math problems, as well as complex scientific calculations.

FEATURES INCLUDE A COLOR MONITOR, 16K RAM MEMORY, BASIC BUILT INTO THE CONSOLE, AND TWO COMPREHENSIVE MANUALS EASY FOR THE BEGINNER - CHALLENGING FOR THE EXPERT.

Programs are sealed securely in SOLID STATE SOFTWARE COMMAND MODULES. These ROM packs actually add memory to the TI-99/4 so that the console's memory can be utilized for user input.

The software applications are presently broken down into four areas: 1) Home Management/Personal Finance, 2) Education, 3) Entertainment, 4) BASIC.

SYO-8994A $1150.00

MICROPROCESSORS	
F80	$16.95
Z80 (2MHz)	$10.95
Z80A (4MHz)	$14.95
CDP1802CD	$19.95
6502	$11.95
6800	$9.75
6802	$14.00
8008-1	$15.95
8035	$24.00
8035-8	$24.00
8080-A	$10.00
8085	$23.00
TMS9900TL	$49.95
8080A SUPPORT DEVICES	
8212	$2.90
8214	$4.65
8216	$2.75
8224 (2MHz)	$4.30
8226	$2.75
8228	$6.40
8238	$6.40
8243	$8.00
8251	$7.50
8253	$20.00
8255	$6.40
8257	$18.00
8259	$18.00
8275	$51.20
8279	$17.70
USRT	
S2350	$10.95
UARTS	
AY5-1013A	$5.25
AY5-1014A	$8.25
TR1602B	$5.25
TMS6011	$5.95
6800 PRODUCT	
6821P	$5.25
6828P	$9.50
6834P	$16.95
6850P	$4.80
6852P	$5.25
6860P	$9.25
6862P	$12.00
6875L	$7.30
6880P	$2.50
CHARACTER GENERATORS	
2513 Upper (1-12+5)	$6.75
2513 Lower (1-12+5)	$6.75
2513 Upper (5 volt)	$9.75
2513 Lower (5 volt)	$10.95
PROMS	
1702A	$5.00
2708	$9.95
2716	$49.95
2716 (5v)	$49.95
2758 (5v)	$30.00
DYNAMIC RAMS	
4115	$5.00
416D/4116 (200ns)	$12.50
2104/4096	$4.00
2107B-4	$3.95
TMS4027/4096	$4.00

STATIC RAMS	1—15	16—100
2114 (450ns)	$8.00	$6.95
2114 (300ns)	$9.00	$8.00
TMS4044/ MM5257(450ns)	$8.00	$7.50
TMS4044/ MM5257(300ns)	$9.95	$8.75
21L02 (450ns)	$1.50	$1.20
21L02 (250ns)	$1.75	$1.50
4200A (200ns)	$9.95	$8.50
410D (200ns)	$8.25	$7.00

JADE SERIAL/PARALLEL I/O BOARD

Features Include: S-100 Compatible ■ Two serial I/O Ports using software-programmable UARTS ■ One general purpose latched parallel I/O Port ideal for printer interface or control ■ Switch selectable ■ On board "Kansas City" standard cassette interface useable upto 1200 Baud, allowing storage of up to 180,000 bytes on a 30 minute audio cassette.

Assembled and Tested IOI-1040A, ... $179.95
Kit IOI-1040K, ... $124.95
Bare Board with Manual IOI-1040B, ... $30.00

SYM-1 $245

Part Number:CPK-5002A

6502- BASED SINGLE BOARD COMPUTER WITH KEYBOARD/DISPLAY, KIM-1 HARDWARE COMPATIBLE, COMPLETE DOCUMENTATION.

SYM-1 CASE, ENX-000005 .... $39.95

THE BIG Z

THE NEW Z-80 CPU BOARD FROM JADE

Features Include: ■ S-100 Compatible, available in 2MHz or 4MHz versions. ■ On-board 2708, 2716, 2516, or 2532 EPROM can be addressed on any 1K, 2K, or 4K boundary, with power-on jump to EPROM. ■ On-board EPROM may be used in SHADOW mode, allowing full 64K RAM to be used. ■ Automatic MWRITE generation if front panel is not used. ■ On-board USART for synchronous or asynchronous RS232 operation (on-board baud rate generator). ■ Reverse-channel capability on USART allows use with buffered peripherals or devices with "not-ready" signal.

2MHz-
Kit: CPU-30200K, 2 lbs ... $149.95
Assembled and Tested: CPU-30200A, 2 lbs ... $199.95

BARE BOARD CPU-30200B....$40.00

4 MHz-
Kit: CPU-30201K, 2 lbs ... $159.95
Assembled and Tested - CPU-30201A, 2 lbs ... $209.95

GRI KEYBOARDS

Features Include: Full 128 character ASCII ■ Tri-Mode Mos encoding ■ MOS/DTL/TTL compatible output ■ Two-key rollover ■ Level and pulse strobe ■ Shift and alpha lock ■ Selectable parity ■ Positive or Negative Logic ■ All new OEM grade components ■ Gold contact, low bounce keyswithces ■ Rugged G-10 printed circuit board ■ Custom 2 shot molded keycaps ■ Low power consumption ■ Optional numeric pad available ■ Custom enclosures available.

Model 756A
Assembled- KBA-30756A $79.95
Model 753K (53 Key, Teletype Keyboard with lower case)
Kit- KBA-30753K $65.95

DB25-CONNECTORS

DB25P- CND1251 .... $2.25
DB25S- CND1252 .... $3.25
DB25C- CND1253 (Cover) $1.50
RS232 Special: DB25P, DB25S, plus Hood only $6.50 (Part Number: CND-1250)

CIRCLE 153 ON READER SERVICE CARD

PRICES GOOD IN THE UNITED STATES AND CANADA

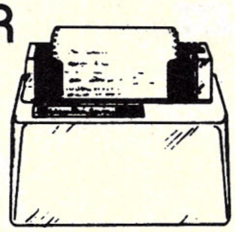
PLACE ORDERS TOLL FREE:

800/421-5809 Continental U.S.
800/262-1710 Inside California

When visiting the Los Angeles area, visit JADE's retail store at 13340 S. Hawthorne Blvd. in the city of Hawthorne

INTEGRAL DATA SYSTEMS MODEL 440 PRINTER
THE PAPER TIGER

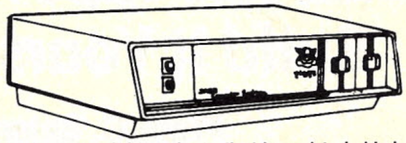
$995.00



- Up to 198 CPS
- 1.75" to 9.5" Adjustable Tractor Feed
- Parallel and Serial Interface
- 96 Character ASCII Set
- 132 columns- 6 or 8 lines/inch
- Eight software selectable Character sizes
- 110, 300, 600, 1200 Baud Rate

FOR THE GRAPHIC OPTION WITH 2K, ADD $199.00 Part Number: PRM-33440

THE PIGGY MAINFRAME



This sleak new mainframe is neatly trimmed to hold six S-100 boards, three mini-floppy drives, and is available in five colors. Power requirements: 115/220 VAC, 50/60 HZ. Weight: 27 lbs. (with drives). Dimensions: 21.375" Wide X 8.4" High X 15.875" Deep. Power Supply: 8 volts at 18 Amps unregulated, + 16 volts at 3 amps unregulated, - 16 volts at 3 amps unregulated, + 5 volts at 3 amps regulated, + 12 volts at 3 amps regulated.

THE PIGGY IS HERE!

THE PIGGY (Without Drives).... $475.00
PART NUMBER: ENS-106320

MEM-2
16 K Static RAM Board

Kit- (450ns)	$250.00
Kit- (250ns)	$285.00
Assembled- (450ns)	$325.00
Assembled- (250ns)	$350.00

DIP SWITCHES

Part No.	positions 1-9		Part No.		
SWD-103	3	$1.18	SWD-107	7	$1.30
SWD-104	4	$1.20	SWD-108	8	$1.34
SWD-105	5	$1.24	SWD-109	9	$1.36
SWD-106	6	$1.28	SWD-110	10	$1.38

DYNAMIC RAM BOARDS EXPANDABLE TO 64K
32K VERSION • KITS

Uses 4115 (8K×1, 250ns) Dynamic RAM's, can be expanded in 8K increments up to 32K

8K	$159.95	24K	$249.95
16K	$199.95	32K	$299.95

64K VERSION • KITS

Uses 4116 (16K×1, 200ns) Dynamic RAM's, can be expanded in 16K increments up to 64K

16K	$249.95	48K	$469.95
32K	$369.95	64K	$569.95

STATIC RAM BOARDS

JADE 8K

Kits:	450ns	$125.95
	250ns	$149.75
Assembled & Tested:	450ns	$139.75
	250ns	$169.75
Bare Board:		$ 25.00

32K Static
Assembled & Tested:

250ns	$795.00
450ns	$725.00
250ns Kit	$575.00

CABLES

MINI-DISK CABLE KIT: To connect two 5 1/4" drives to disk controller board. Contains assembled and tested 5' long signal cable with 34 pin edge connectors. Also includes cables and connector for D.C. power supply.
WCA-3431K $34.95

8" DISK CABLE KIT: To connect two 8" disk drives to edge-type controller (e.g., Versafloppy, Double-D). Contains assembled and tested signal cable with connectors plus cable and connectors for both A.C. and D.C. power.
WCA-5031K $38.45

8" DISK CABLE KIT: Same as WCA-5031K except controller end of signal cable uses "Header" type connector, e.g., for Tarbell Controller.
WCA-5032K $38.95

SIGNAL CABLE ONLY: For one 5 1/4" drive to edge type controller connector (e.g., TRS-80 to Vista Disk Drive).
WCA-3421A $24.95

Same as Above, except for two 5 1/4" drives. WCA-3431A $29.95

SPECIAL

5¼" Diskettes
$29.95/Box of TEN
SPECIFY SOFT, 10, or 16 SECTOR

8" SINGLE SIDE, SINGLE DENSITY
$34.95/Box of TEN

VB-1B VIDEO INTERFACE

S-100 Compatible Serial Interface with Sockets Included.

Kit	$117.95
Assembled & Tested	$159.95
Bare Board w/manual	$ 35.00

FLOPPY DISK INTERFACE

JADE FLOPPY DISK (Tarbell board)

JADE KIT	**$190.00**
Assembled & Tested	**$260.00**

S.D. Computer Products VERSA-FLOPPY

Kit	$159.95
Assembled & Tested	$239.00

L.S. Engineering EPROM Eraser

Features Include: Erases up to 48 I.C.s at a time • Uses popular (and readily available 12" shortwave UV bulbs) • Handsome smoke-plastic case • Includes bulb • Just plug it in and you're ready to go!

XME-3200, ... $39.95

JADE FLOPPY DISK SPECIAL

INCLUDES: ■ Two Siemens/GSI 8" Floppy Disk Drives ■ Power Supply ■ Jade Double Density Board (Kit) ■ CP/M™ Operating System with Basic E (CP/M is a registered trademark of Digital Research) ■ Package of 10 Blank 8" Diskettes (Double Density) ■ Interface Cables.

If purchased seperately this package would cost: $1544.95.

JADE SPECIAL $1225.00

THE JADE "DOUBLE-D" INTELLIGENT DISK CONTROLLER

- READ WRITE IN SINGLE OR DOUBLE DENSITY
- 8" or 5¼" DRIVES
- CP M COMPATIBLE IN EITHER DENSITY
- ON-BOARD Z80 CPU ALLOWS UNIVERSAL SYSTEM COMPATIBILITY
- PROGRAMMED DATA TRANSFER - NO DMA
- CONTROLS UP TO 8 DRIVES
- SOFTWARE SELECTABLE DENSITY

Our new controller utilizes the IBM standard formats for proven reliability. Data recovery is enchaced through use of a phase-locked-loop data seperation circuit and write precompensation. Single and double density diskettes can be mixed in the same system.

Assembled and Tested
IOD-1200A, 2 lbs. . . $299.95
Kit
IOD-1200K, 2 lbs. . . $249.95
Bare Board w/ Manual
IOD-1200B, 2 lbs. . . . $55.00
Manual
IOD-1200M, 1 lb. . . . $10.00

DISK DRIVES

MPI B51 5 1/4" MSM—155100 $295.00
Single or double density, up to 40 tracks, track to track access time only 5ms.

MPI B52 5 1/4" MSM-155200 $450.00
Double head version of MPI B51

Shugart SA400 5 1/4" $325.00
Single Density, 35 Track

Siemens FDD100-8 8" $495.00
Shugart 801R replacement, Single or double density, runs cooler and quieter.

Siemens FDD200-8 8" $575.00
Double head version of FDD100-8 capable of double density, double sided storage.

Shugart 801R 8" $575.00
Hard or soft sectored, 400K BYTE drive.

Vista V80 Mini Disk SYSTEM FOR TRS—80
$395.00

PART NUMBER: MSM-358000

Includes disk drive, power supply, regulator board, and compact case. The V-80 offers 23% more storage capacity. Simply take it out of the box, plug in the cable, and it's ready to run. Requires 16K, Level II, expansion interface.

Interface Cable .. $24.95
PART NUMBER: WCA-3421A

POWER SUPPLIES

PSD-249A For a Single 5 1/4 Disk Drive By Power-One or Alpha Power +5V at.7A, +12V at 1.1A .. $52.00

PSD-205A For Single 8 Disk Drive By Power-One 5V at 1A, -5V at 5A, 24V at 1.5A $89.95

PSD-206A For Two 8 Disk Drives By Power-One or Alpha Power 5V at 2.5A, -5V at .5A -24V at 3A $125.00

Rockwell Aim-65 Power Supply PSX-030A $59.95

KIM-1 or SYM-1 Power Supply PSX-020A $59.95

JADE ISO—BUS MOTHERBOARDS

Comes in either 6, 12, or 18 slot sizes. These boards with a special ground plane assure a silent operation

JADE 6 Slot

Kit	$49.95
Assembled	$59.95
Bare Board	$24.95

JADE 12 Slot

Kit	$89.95
Assembled	$99.95
Bare Board	$39.95

JADE 18 Slot

Kit	$129.95
Assembled	$149.95
Bare Board	$59.95

JADE Computer Products

4901 W. ROSECRANS AVE., HAWTHORNE, CALIFORNIA, 90250
(213) 679-3313

ORDER TOLL FREE

800-262-1710 INSIDE CALIFORNIA
800-421-5809 CONTINENTAL U.S.

WRITE FOR OUR FREE CATALOG

Cash, checks, money orders, and credit cards accepted. Minimum order: $10.00, California residents add 6% sales tax. Minimum shipping and handling charge: $2.50. Discounts available at OEM quantities.

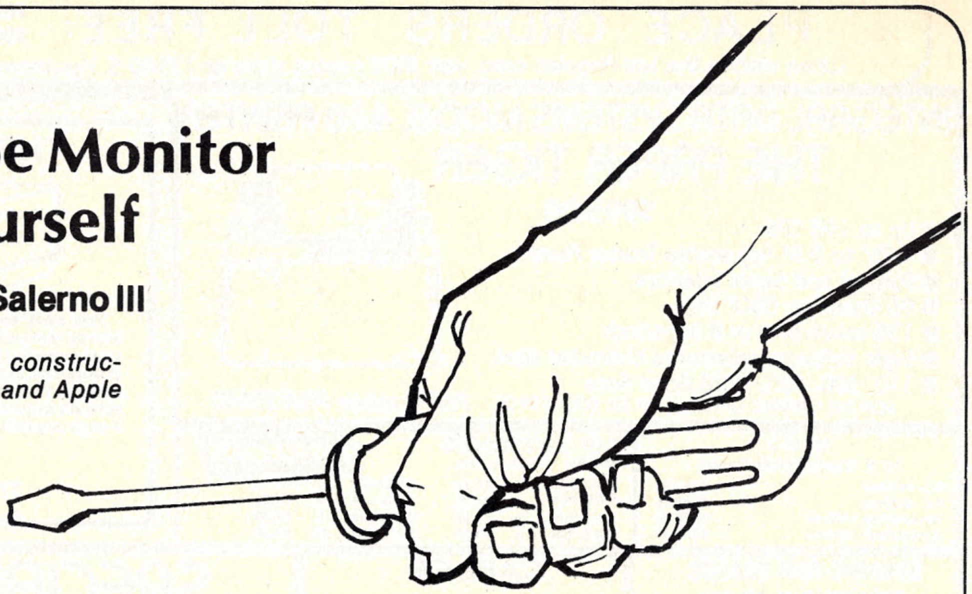
PRICES SUBJECT TO CHANGE WITHOUT PRIOR NOTICE

For Export information send for a JADE INTERNATIONAL CATALOG

Cassette Tape Monitor Build It Yourself

Frank C. Salerno III

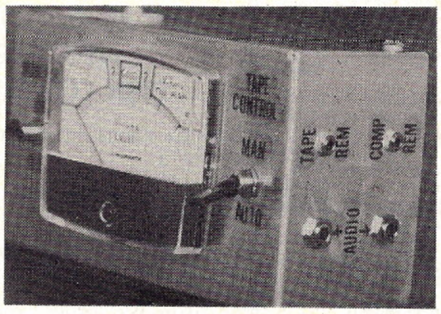
A build-it-from-scratch construction project for TRS-80 and Apple cassette systems.



Here's a handy accessory for your computer if you load programs from an inexpensive cassette recorder. This neat little gadget will: allow monitoring of your cassette recorder's output without pulling out the earphone plug; indicate the correct volume level for loading programs; and, permit manual control of your cassette recorder's motor (if it is under computer control) without pulling out the remote plug.

The unit shown in Photo 1 will work with a TRS-80 computer (with tape control). I am presently using a unit without the tape control feature on my Apple II computer.

Either version can be built in several evenings from readily available parts. Most of the parts can be purchased at your neighborhood Radio Shack store for under $25.00.



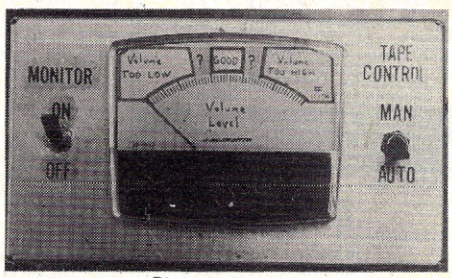


Photo 1. The Unit.

The only parts not available at Radio Shack are the speaker grill, epoxy, dry transfer lettering and clear lacquer.

The Circuit

The circuit schematic is shown in Figure 1. This is the circuit to be used with a computer such as the TRS-80. Deleting J3, J4 and S2 will give a circuit diagram for use with a computer such as the Apple II.

Audio from the tape recorder is applied to either J1 or J2. The other jack is then connected to the computer with a six-foot long miniplug-to-miniplug patch cord. SPK, R1, S1 and R2 comprise the monitor circuit. R2 is the monitor level adjust. R1 is included to prevent circuit impedance from changing if the monitor is switched off. The meter movement, D1, D2, D3 and R3 comprise the volume level meter circuit. D1 and D2 protect the meter movement in case the volume level is too high. R3 is the meter level adjustment. D3 rectifies the audio so the meter can display it.

The tape control circuit is comprised on J3, J4 and S2.

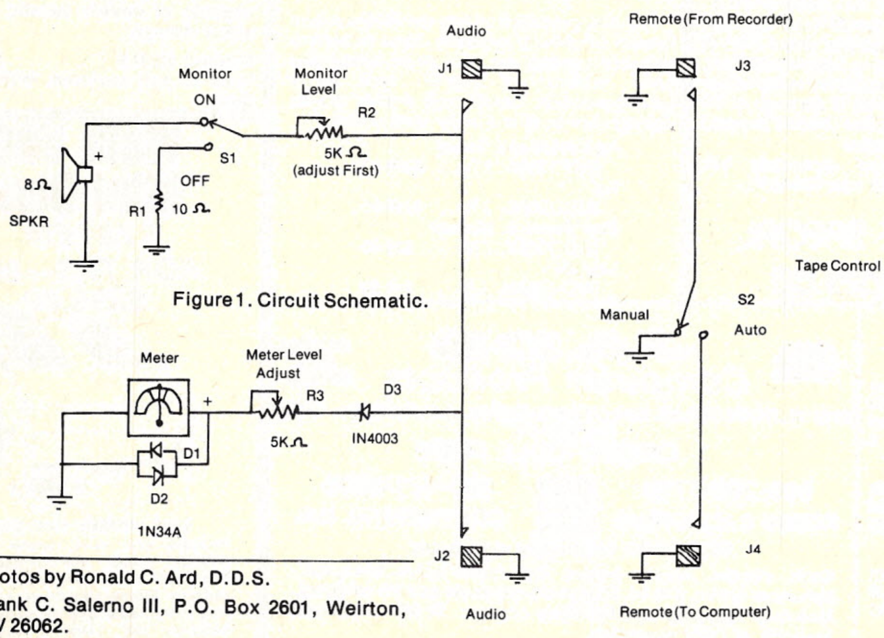


Figure 1. Circuit Schematic.

Photos by Ronald C. Ard, D.D.S.

Frank C. Salerno III, P.O. Box 2601, Weirton, WV 26062.

Cassette, con't...

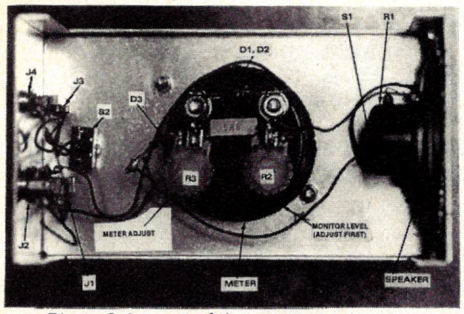


Photo 2. Layout of the components.

Construction

Layout of the components is not critical. If the general layout in Photo 2 is followed there should be no problems. The only tight spot is between S1 and the speaker magnet. This was solved by wrapping some plastic electrical tape around the speaker magnet before mounting it.

A patch cord has to be constructed from a piece of speaker wire and a sub-mini plug on each end. This patch cord connects the tape control circuit to the tape recorder via J3.

Mark a 1¾ inch circle on the side of the case on which the speaker is to be mounted. Drill a pilot hole big enough for a coping saw blade to fit through. Put a coping saw blade through the pilot hole, attach it to the saw and saw away until the circle is cut out. File any rough edges. Cut a piece of speaker grill mesh 1-15/16 inch by 2-3/8 inch. Check to see that it covers the hole neatly from inside

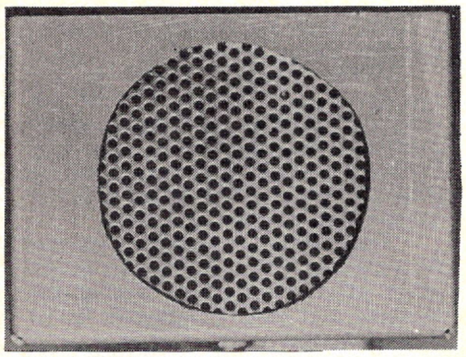


Photo 3. Side view of speaker grill.

the case (Photo 3). Next, follow the drilling guide that comes with the meter movement. Cut an opening for the meter using the same method as the speaker. Mark the opening to be cut, drill a pilot hole, and use the coping saw. File any rough edges. See if the meter fits properly in the hole that was just cut. Do not mount it yet. There is more drilling to be done and it might get damaged.

Now mark the locations for S1 and S2. Remember, leave S2 out if the unit is to be used with an Apple II. Mark holes ¾ inch from the edge of the case and 1½ inch from the top. Drill holes to fit your switches. Check for proper fit, but don't mount them.

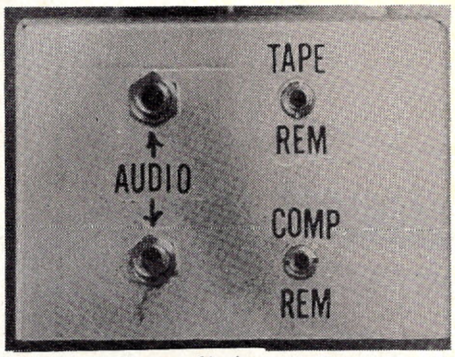


Photo 4. Side view of jacks.

Mark and drill holes for J1, J2, J3 and J4 as shown in Photo 4. Remember, you only need J3 and J4 if this is for a TRS-80. Mount the switches and jacks in their proper locations. Solder the wires and the 10 ohm resistor to S1.

Take the 2 inch speaker and wrap electrical tape around the speaker magnet. Using epoxy, glue the piece of grill and speaker in place. Make sure the speaker terminals are in a convenient position. Put the case in a safe place and allow the epoxy to harden.

While the epoxy is hardening, the meter scale and label can be copied and applied. The scale (shown full size in Figure 2) can be copied on a 1 inch by 3 inch self-stick label and trimmed to size. The scale can be colored if desired. The new meter function label, "Volume Level," can also be copied onto a self-stick label.

After the new scale and function label have been copied the meter cover must be opened carefully by using a small screwdriver. Four slots in the meter case make removal of the cover quite simple. Apply the new scale and function label (referring to Photo 1) and snap the meter together again.

If the epoxy has hardened, the meter and other components can be mounted and wired according to Figure 1 and Photo 2. Don't close the case or apply the dry transfer lettering when the wiring is finished since R2 and R3 still need adjusting.

Checkout and Calibration

Hook up the unit using the new patch cords. Load a program from the cassette recorder as you normally would. Turn the monitor switch on. Adjust R2 until the volume is at a comfortable listening level. Adjust R3 at this time until the meter needle is centered in the GOOD range. Check for proper operation of the switches. If all is in order disconnect the unit and close the case. The dry transfer lettering can now be applied. Once the lettering has been applied a thin coat of clear lacquer should be brushed over them to make them scratch resistant.

Using the Unit

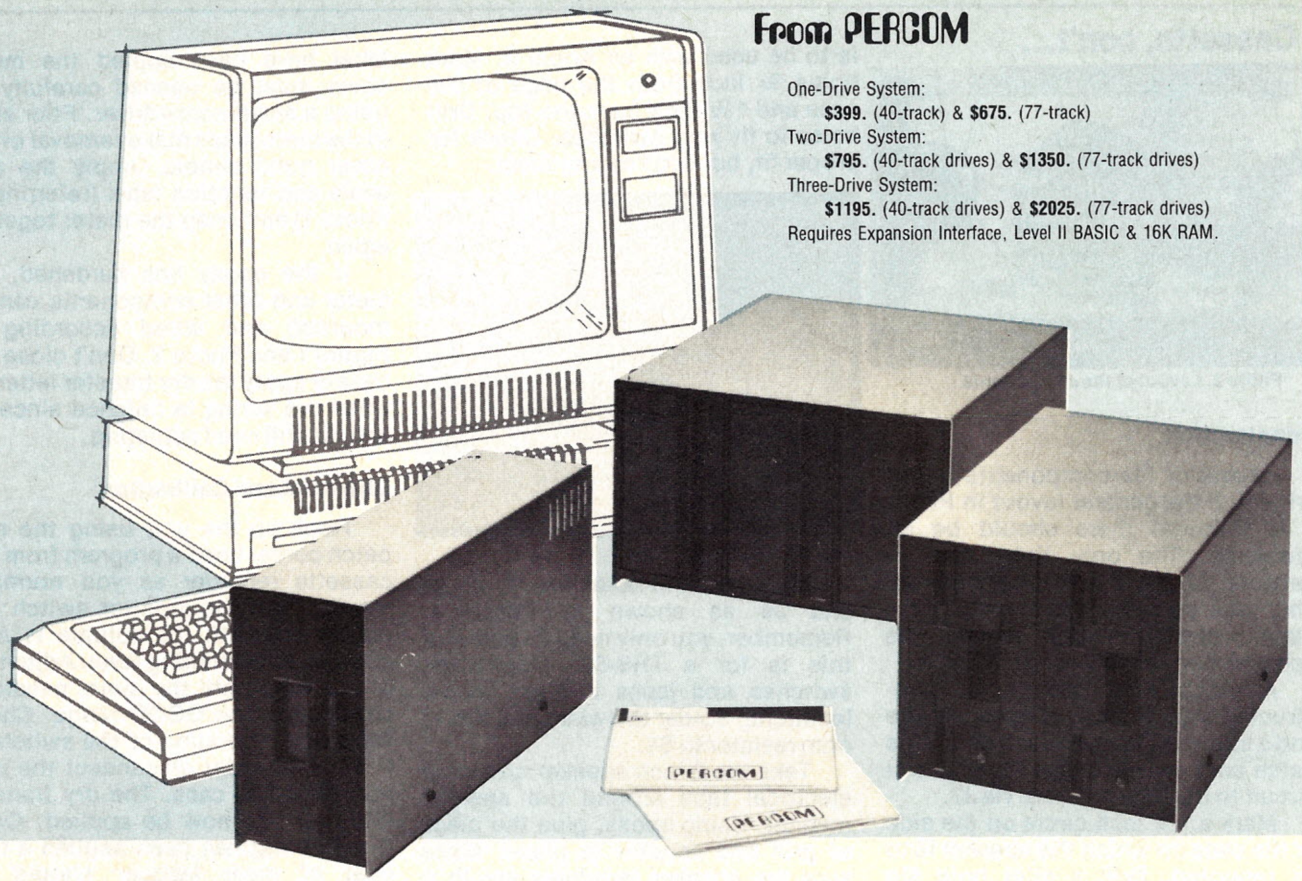
Reconnect the unit. Whenever a program is loaded from the cassette recorder simply adjust the recorder's volume level until the meter is in the GOOD range. The tape control and monitor switches save you from ever pulling a mini-plug to hear or control what is being loaded. □

Parts List

Part	Description	Radio Shack Catalog #	Qty.
Case	Aluminum Case, 5¼ x 3 x 2-1/8	270-238	1
Meter	0-1 mA Panel Meter	22-052	1
D1, D2	Germanium Diodes, 1N34A	276-1123	2
D3	Diodes, 1N4003	276-1102	1
S1, S2	SPDT Toggle Switches	275-662	2
R1	10 Ohm Resistor	271-001	1
R2, R3	5000 Ohm Trim Potentiometers	271-217	2
J1, J2	Mini-phone Jacks	274-253	2
J3, J4	Sub-Mini Phone Jacks	274-292	2
SPKR	2 inch Speaker, 8 Ohm	40-245	1
Cord	6 ft. Patch Cord Mini-to-Mini	42-2420	1
MISC	Epoxy, Clear Lacquer, Self-Stick Labels, Dry Transfer Letters, Wire, and Speaker Grill		

From PERCOM

One-Drive System:
$399. (40-track) & **$675.** (77-track)
Two-Drive System:
$795. (40-track drives) & **$1350.** (77-track drives)
Three-Drive System:
$1195. (40-track drives) & **$2025.** (77-track drives)
Requires Expansion Interface, Level II BASIC & 16K RAM.



Low Cost Add-On Storage for Your TRS-80*. In the Size You Want.

When you're ready for add-on disk storage, we're ready for you. Ready with six mini-disk storage systems — 102K bytes to 591K bytes of additional *on-line* storage for your TRS-80*.

- **Choose either 40-track TFD-100™ drives or 77-track TFD-200™ drives.**
- **One-, two- and three-drive systems immediately available.**
- **Systems include Percom PATCH PAK #1™, on disk, at no extra charge. PATCH PAK #1™ de-glitches and upgrades TRSDOS* for 40- and 77-track operation.**
- **TFD-100™ drives accommodate "flippy disks." Store 205K bytes per mini-disk.**
- **Low prices. A single-drive TFD-100™ costs just $399. Price includes PATCH PAK #1™ disk.**
- **Enclosures are finished in system-compatible "Tandy-silver" enamel.**

Whether you need a single, 40-track TFD-100™ add-on or a three-drive add-on with 77-track TFD-200™s, you get more data storage for less money from Percom.

Our TFD-100™ drive, for example, lets you store 102.4K bytes of data on one side of a disk — compared to 80K bytes on a TRS-80* mini-disk drive — and 102.4K bytes on the other side, too. Something you can't do with a TRS-80* drive. That's almost 205K bytes per mini-disk.

And the TFD-200™ drives provide 197K bytes of on-line storage per drive — 197K, 394K and 591K bytes for one-, two and three-drive systems.

PATCH PAK #1™, our upgrade program for your TRSDOS*, not only extends TRSDOS* to accommodate 40- and 77-track drives, it enhances TRSDOS* in other ways as well. PATCH PAK #1™ is supplied with each drive system at no additional charge.

The reason you get more for less from Percom is simple. Peripherals are not a sideline at Percom. Selling disk systems and other peripherals is our main business — the reason you get more engineering, more reliability and more back up support for less money.

In the Product Development Queue . . . a *printer interface* for using your TRS-80* with any serial printer, and . . . the *Electric Crayon*™ to map your computer memory onto your color TV screen — for games, animated shows, business displays, graphs, etc. Coming PDQ!

™ TFD-100, TFD-200, PATCH PAK and Electric Crayon are trademarks of PERCOM DATA COMPANY.
*TRS-80 and TRSDOS are trademarks of Tandy Corporation and Radio Shack which have no relationship to PERCOM DATA COMPANY.

PERCOM

PERCOM DATA COMPANY, INC.
211 N. KIRBY • GARLAND, TX. • 75042

To order add-on mini-disk storage for your TRS-80*, or request additional literature, call Percom's toll-free number: 1-800-527-1592. For detailed Technical information call (214) 272-3421.

Orders may be paid by check or money order, or charged to Visa or Master Charge credit accounts. Texas residents must add 5% sales tax.

Percom *'peripherals for personal computing'*

CIRCLE 177 ON READER SERVICE CARD

BASIC Tricks

Jordan Mechner

It's a familiar situation. You're sitting down at your personal computer, happily typing in a program you found in a book or magazine. Upon running the program, you're hit with a loud beep and a SYNTAX ERROR. You look at the last line you typed, and find something like

```
1040 FOR I = 1 TO 8 :
FOR J = 1 TO 8 : A(I,J) = 0 :
NEXT J : NEXT I
```

What's the problem? Your BASIC doesn't support arrays of more than one dimension. Or perhaps it's string functions, or formatted print capability, or some other advanced feature.

Whatever you do, don't throw up your hands in despair and turn off the computer! There are ways to simulate these advanced features even in minimal BASIC.

Double subscripts

Every array of two dimensions can be "unraveled" into an array of one dimension. For example, the array

```
1  2  3  4
5  6  7  8
9 10 11 12
```

can be "unraveled" into

```
1 2 3 4 5 6 7 8 9 10 11 12
```

What we want is a formula that will tell us what position in the second array corresponds to a position in the first array. The formula can be easily found by inspection. If A1 is the first array, A2 is the second array, and C is the number of columns in A1, then

```
A1(I,J) = A2(C*I + J-C)
```

Wherever you encounter an expression using double subscripts, this formula can be put to good use. Checkers, Nim, Tic-Tac-Toe and other games requiring arrays of two dimensions can be programmed even in minimal BASIC.

Formatting

There is a simple method for controlling the number of decimal places printed by the computer. To print the number N to P decimal places, use the formula

```
INT((10 ^ P)*N)/(10 ^ P)
```

Jordan Mechner, 85 Haights Cross Rd., Chappaqua, NY 10514.

The effect of this formula is to move the decimal point forward P place, to chop off the remaining places with the INT function, and to move the decimal point back P places to where it started.

This formula cannot match the power of the PRINT USING statement, which allows automatic printing of commas, dollar signs, and special symbols, but for most applications it is sufficient.

Trigonometric functions

The SIN(), COS() and TAN() functions can be calculated by fairly simple subroutines. Here is the SIN() subroutine.

```
10 S = 0 : I = 1
20 FOR C = 1 TO N STEP 2 :
F = 1 : FOR D = 1 TO C :
F = F*D : NEXT D :
S = S + I*(X ^ C)/F : I = -I :
NEXT C
30 RETURN
```

S represents the value of SIN(X). N represents the precision with which S is to be calculated (the greater the value of N, the more accurate the value of S). I, C, D and F are important only within the subroutine.

The subroutine for COS() is similar. The only difference is that line 20 must be changed to

```
20 FOR C = 0 TO N STEP 2 ...
```

And, since tan x = sin x/cos x, the value of the TAN() function can be calculated by using the above subroutines to find the value of the SIN() function and the COS() function and dividing the two results.

Rounding off

The expression

```
INT (X + 0.5)
```

rounds X off to the nearest integer. For example,

```
INT (0.7 + 0.5) = INT (1.2) = 1
INT (0.4 + 0.5) = INT (0.9) = 0
```

In short, if X is closer to the integer following it than to the one preceding it, the 0.5 will send it over the top.

We can use this trick to improve our "formatting" expression to

```
INT ((10 ^ P)*N + 0.5)/(10 ^ P)
```

The STEP clause

Many extended BASICs support the FOR-TO-STEP statement. This useful clause can be simulated with the addition of only one extra line.

```
10 FOR V = A TO B STEP S
...
...
20 NEXT V
```

can be replaced by

```
10 FOR U = A TO INT(B/S) + 1
15 V = S*U-S
...
20 NEXT U
```

If the increment S is negative, however, this will not work. Instead,

```
10 FOR V = B TO A STEP -1
...
20 NEXT V
```

can be replaced by

```
10 FOR U = A TO B
15 V = (A + B)-U
...
20 NEXT U
```

The ELSE clause

A statement of the form

```
10 IF C THEN A ELSE B
```

is equivalent to

```
10 IF C THEN A : GO TO 30
20 B
30 ...
```

This is a rather obvious modification, but it's useful nonetheless.

Logarithms

Your BASIC may support a natural logarithm function, but not a common logarithm or base-n logarithm function. With any logarithm function, however, you have the ability to calculate logs to any base.

The base-n logarithm of x is, for any n,

```
LOG (X)/LOG(N).
```

Roots

Nearly all BASICs have a SQR() function, but not many have functions to calculate other roots, such as cube roots or fourth roots. There is a simple formula to do this.

The nth root of x is, for any x,

```
X (1/N).
```

Using this formula, the square root of 7 is

```
7 ^ 0.5
```

and the cube root of 27 is

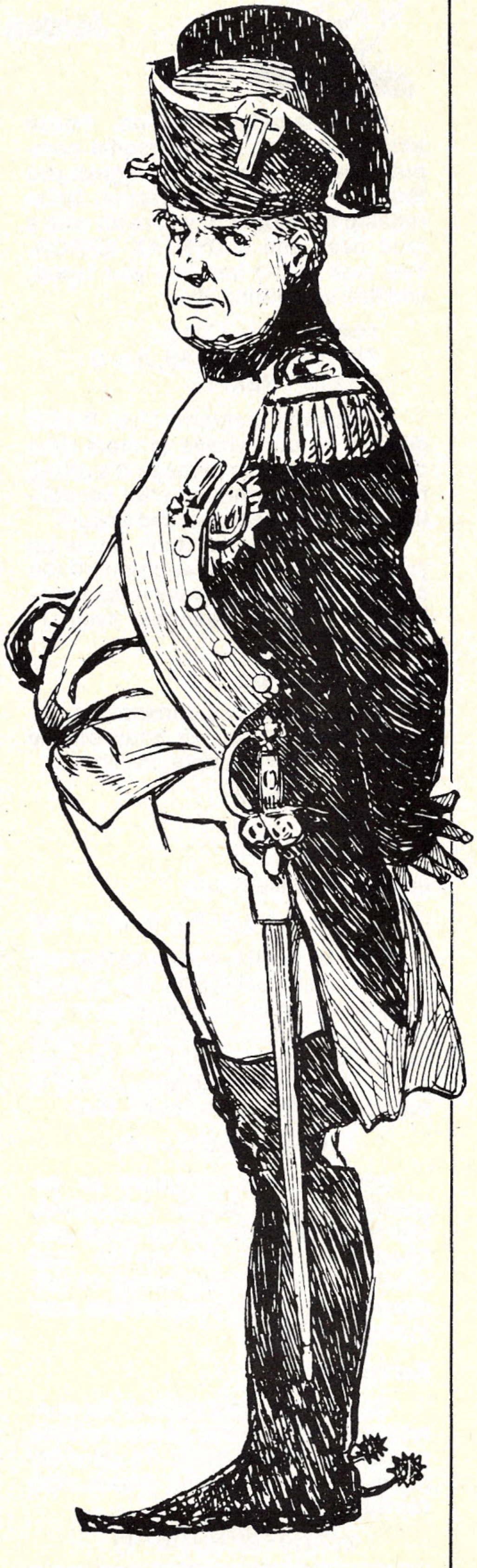
```
27 ^ 0.333333.
```

□

Another new game from Creative Computing ....

The French Military Game

Gerald H. Herd



This article describes a game with fairly simple rules which is surprisingly non-trivial. The game is called the French Military Game, or FMG for short. FMG is analyzed in some detail in **The Sixth Book of Mathematical Games from Scientific American** by Martin Gardner.

The Game

The playing field for FMG is shown in Figure 1. For notational reasons, the spots or circles on the playing field are numbered. One player, White, has three men that are initially placed on spots 1, 2, and 4.

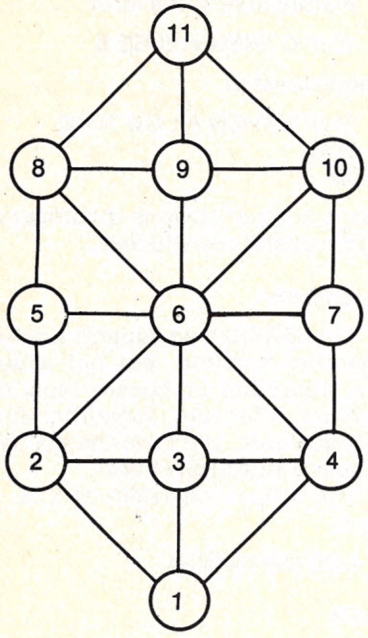
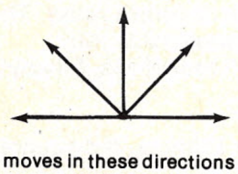


Figure 1. FMG Playing Field.



White moves in these directions only.

Black, his opponent, has only one man, placed initially on spot 6. White has the first move and the game proceeds with White and Black alternating moves. Black may move his piece in any direction along a line from one spot to a vacant neighboring spot, such as spot 6 to spot 10. White, on his turn, may also move any one of his pieces along a line to a vacant neighboring spot, but only left, right, forward, diagonally left or diagonally right forward, never backwards (toward spot 1). There are no captures and two pieces may never occupy the same spot. White wins the game by pinning the Black piece so that Black has no move. This usually occurs with the White pieces on spots 8, 9, and 10 and the Black piece pinned on spot 11. It is possible to pin Black on spot 5 or spot 7, but Black has to play rather stupidly to let this happen. Black wins by reaching spot 1 or if a situation develops in which the same moves are repeated endlessly. White can always win if he plays correctly and does not make any mistakes. White can always win even if the Black piece does not start out on spot 6. The reader should refer to Mr. Gardner's original article for more detailed analysis of the game.

Computer Version

The rules for the computer version of FMG are as described previously. The computer plays the Black piece and keeps up with where the pieces are on the board. The program will ask the White player (you) for a move after printing out the current positions of the pieces. If the White player wishes to concede defeat, he enters a 0, 0 as his move. Otherwise, he moves the

Gerald H. Herd, 3781 Whispering Pines Dr., Pensacola, FL 32504.

piece he desires by entering the starting spot and ending spot for the piece. For instance, White's first move might be entered as 1, 3 to move the White piece on spot 1 to spot 3. The program is set up to assume a victory for Black if Black has not been trapped after 20 moves, as well as check for Black reaching spot 1 or being pinned.

The program is organized as follows. Lines 1 through 390 are initialization routines for the board position and data arrays used by the program. Lines 500 through 840 form the main game playing loop. This loop calls subroutines in the proper order to accept White's move, check it for legality, pick Black's response, detect an end of game, etc. The subroutine at line 2000 checks for a legal move by White. The subroutine at line 3000 calculates a position number which is used by subroutine 4000 to pick Black's best reply. The subroutine at 6000 updates the array A between games so that the program appears to learn to play better as it gains experience.

The learning strategy used is usually referred to as a matchbox learning strategy, after the title of another article by Martin Gardner. Those who wish to read more of the technique are referred to **The Unexpected Hanging** by Martin Gardner. In general, the matchbox learning technique has been useful in programming comparatively simple games such as Nim, Chomp, Tic-Tac-Toe, and FMG.

When programming FMG, I tried to stay as close to Dartmouth BASIC as I could in order to avoid portability problems when running the program on another system. There are two lines of the program which require a little explanation. Line 190 reads data from a floppy disk into array A. Line 830 stores the updated array A back to the disk when the playing session is over. These two lines save what the computer has learned from one playing session to another. For those whose system does not allow data files, replace line 190 with a MAT A=ZER, and line 830 with a REM statement.

A sample game is shown in Figure 2. The program listing and variable cross-reference list is shown in Figure 3.

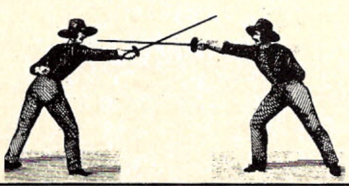
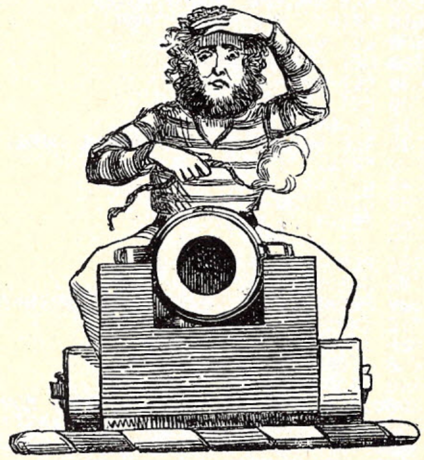


Figure 2. Sample game, where the computer won.

```
?RUN

RUN
WHITE HAS PIECES AT 1      2      4     ///THE BLACK PIECE IS AT  6
YOUR MOVE, FROM-TO X,Y?
?1,3
I MOVE TO  10
WHITE HAS PIECES AT 3      2      4     ///THE BLACK PIECE IS AT  10
YOUR MOVE, FROM-TO X,Y?
?3,6
I MOVE TO  7
WHITE HAS PIECES AT 6      2      4     ///THE BLACK PIECE IS AT  7
YOUR MOVE, FROM-TO X,Y?
?2,5
I MOVE TO  10
WHITE HAS PIECES AT 6      5      4     ///THE BLACK PIECE IS AT  10
YOUR MOVE, FROM-TO X,Y?
?4,7
I MOVE TO  9
WHITE HAS PIECES AT 6      5      7     ///THE BLACK PIECE IS AT  9
YOUR MOVE, FROM-TO X,Y?
?7,10
I MOVE TO  8
WHITE HAS PIECES AT 6      5      10    ///THE BLACK PIECE IS AT  8
YOUR MOVE, FROM-TO X,Y?
??10,9
I MOVE TO  11
WHITE HAS PIECES AT 6      5      9     ///THE BLACK PIECE IS AT  11
YOUR MOVE, FROM-TO X,Y?
?5,8
I MOVE TO  10
WHITE HAS PIECES AT 6      8      9     ///THE BLACK PIECE IS AT  10
YOUR MOVE, FROM-TQ X,Y?
?6,7
I MOVE TO  6
WHITE HAS PIECES AT 7      8      9     ///THE BLACK PIECE IS AT  6
YOUR MOVE, FROM-TO X,Y?
?0,0
I WIN...
WANT TO PLAY AGAIN?(1=YES,0=NO)
```

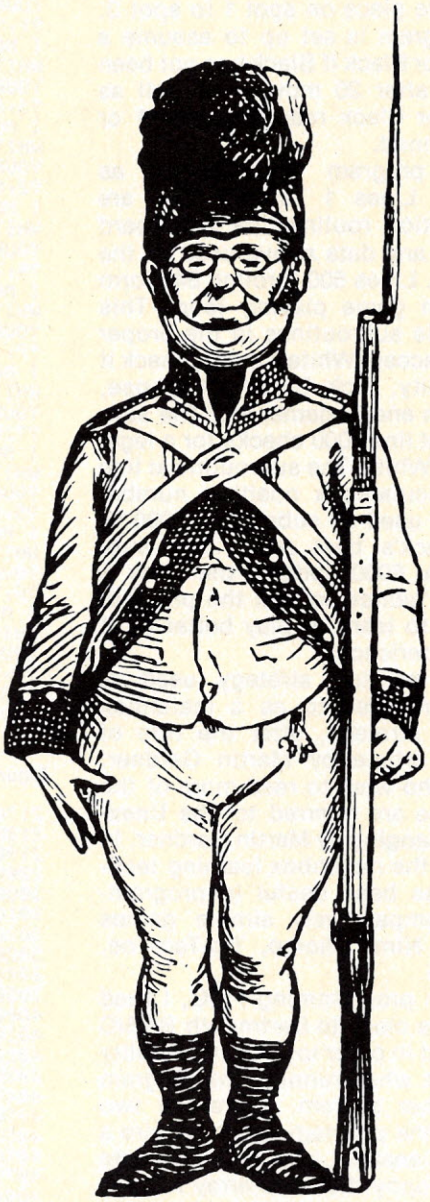
Figure 3A. Listing for FMG.

```
2170 REM LEGAL MOVE FOR WHITE
2180 LET W[J2]=Y
2190 LET L2=+1
2200 RETURN
3000 REM SUB. TO CALCULATE NEW WHITE POSITION NUMBER
3010 LET A3=FNT(W[1])+FNT(W[2])+FNT(W[3])
3020 FOR S3=1 TO 165
3030 IF S[S3]=A3 THEN 3050
3040 NEXT S3
3050 RETURN
4000 REM PICK BLACKS BEST MOVE
4010 LET M4=0
4020 FOR I4=1 TO 11
4030 IF L[B,I4]=0 THEN 4110
4040 REM ADJACENT MOVE FOUND, OCCUPIED BY WHITE?
4050 FOR J4=1 TO 3
4060 IF W[J4]=I4 THEN 4110
4070 NEXT J4
4075 IF M4=0 THEN 4100
4080 IF A[S3,M4] >= A[S3,I4] THEN 4110
4090 REM NEW BEST MOVE
4100 LET M4=I4
4110 NEXT I4
4120 RETURN
6000 REM UPDATE AT GAME END
6010 FOR I6=1 TO I0-1
6020 LET S=R[I6,1]
6030 LET M=R[I6,2]
6040 LET A[S,M]=A[S,M]+W
6050 NEXT I6
6060 RETURN
8999 REM DATA FOR L ARRAY
9000 DATA 0,2,2,2,0,0,0,0,0,0,0
9010 DATA 1,0,2,0,2,2,0,0,0,0,0
9020 DATA 1,2,0,2,0,2,0,0,0,0,0
9030 DATA 1,0,2,0,0,2,2,0,0,0,0
9040 DATA 0,1,0,0,0,2,0,2,0,0,0
9050 DATA 0,1,1,1,2,0,2,2,2,2,0
9060 DATA 0,0,0,1,0,2,0,0,0,2,0
9070 DATA 0,0,0,0,1,1,0,0,2,0,2
9080 DATA 0,0,0,0,0,1,0,2,0,2,2
9090 DATA 0,0,0,0,0,1,1,0,2,0,2
9100 DATA 0,0,0,0,0,0,0,2,2,2,0
9110 END
```



Military, con't...

Figure 3A. Listing for FMG



```
5 DEF FNT(X)=2↑(X-1)
10 REM FRENCH MILITARY GAME, SIXTH BOOK OF MATH.GAMES FR.SCI.AM.-GARDNER
20 REM INCORPORATES MATCH.LEARNING FROM UNEXPECTED HANGING-GARDNER
25 REM PROGRAMMED BY G.H. HERD   SEPT 1978
29 REM DECLARE ARRAYS S,A,W,R,L INTEGER TO SAVE STORAGE SPACE:
30 DIM L[11,11],S[165],R[30,2],A[165,11],W[3]
40 REM L=LEGAL MOVE ARRAY,S()=ALL POSS. LEGAL WHITE POSITIONS
50 REM W(I)=POSITION OF WHITE PIECE.R=RECORD OF MOVES
60 REM COMPUTER MOVES BLACK PIECE, HUMAN WHITE
70 REM GEN. S ARRAY
75 LET I=1
80 FOR L=1 TO 9
90 LET L1=FNT(L)
100 FOR M=L+1 TO 10
110 LET M1=FNT(M)
120 FOR R=M+1 TO 11
130 LET S[I]=L1+M1+FNT(R)
140 LET I=I+1
150 NEXT R
160 NEXT M
170 NEXT L
180 REM LOAD UP LEARNING CURVE
190 LOAD  DATA #5,34,A
200 FOR I=1 TO 30
210 FOR J=1 TO 2
220 LET R[I,J]=0
230 NEXT J
240 NEXT I
250 FOR I=1 TO 11
260 FOR J=1 TO 11
270 READ L[I,J]
280 NEXT J
290 NEXT I
300 REM L(I,J)=0 MEANS MOVE I TO J NOT LEGAL
310 REM L(I,J)=1 MEANS MOVE I TO J LEGAL BLACK ONLY
320 REM L(I,J)=2 MEANS MOVE I TO J LEGAL BOTH WHT. AND BLK.
330 REM
340 LET B=6
350 LET W[1]=1
360 LET W[2]=2
370 LET W[3]=4
380 LET I0=1
390 GOSUB 3000
500 REM MAIN PROGRAM LOOP
510 PRINT "WHITE HAS PIECES AT";W[1];W[2];W[3]"///";
520 PRINT "THE BLACK PIECE IS AT ";B
530 PRINT "YOUR MOVE, FROM-TO X,Y?"
540 INPUT X,Y
541 IF X=0 AND Y=0 THEN 730
550 REM WAS IT A LEGAL MOVE
560 GOSUB 2000
570 IF L2=1 THEN 600
580 PRINT "FOUL!!!!!! TRY AGAIN."
590 GOTO 510
600 REM LEGAL MOVE, WHAT IS WHITE POSITION INDEX (S3)?
610 GOSUB 3000
615 IF I0 <= 19 THEN 630
616 PRINT "THAT'S 20 MOVES ACE, AND YOU HAVEN'T TRAPPED ME!"
617 GOTO 730
620 REM WHAT IS BLACKS BEST MOVE, IF ANY
630 GOSUB 4000
640 IF M4=0 THEN 760
650 REM MACHINE MAKES MOVE:
660 PRINT "I MOVE TO ";M4
670 LET R[I0,1]=S3
680 LET R[I0,2]=M4
690 LET B=M4
700 LET I0=I0+1
710 IF B#1 THEN 500
720 REM HERE ONLY IF MACHINE WINS:
730 PRINT "I WIN..."
740 LET W=1
750 GOTO 780
760 PRINT "I LOST..."
770 LET W=-1
780 REM UPDATE ARRAY A
790 GOSUB 6000
800 PRINT "WANT TO PLAY AGAIN?(1=YES,0=NO)"
810 INPUT Z
820 IF Z=1 THEN 340
830 STORE  DATA #5,34,A
840 STOP
2000 REM SUBR. TO CHECK WHITES MOVE FOR LEGALITY
2010 REM A MOVE IS LEGAL IF STARTING POSITION  HAS A WHITE PIECE ON IT,
2020 REM AND L(X,Y) SAYS MOVE IS LEGAL FOR WHITE, AND BLACK NOT AT Y
2030 LET L2=-1
2040 FOR I2=1 TO 3
2050 LET J2=I2
2060 IF W[I2]=X THEN 2100
2070 NEXT I2
2080 REM NO  WHITE PIECE AT X
2090 GOTO 2200
2100 REM IF BLACK OCCUPIES Y THEN ILLEGAL
2110 IF B=Y THEN 2200
2120 REM ILLEGAL IF WHT PIECE ALREADY AT Y
2130 FOR I2=1 TO 3
2140 IF W[I2]=Y THEN 2200
2150 NEXT I2
2160 IF L[X,Y]#2 THEN 2200
```

Military, con't...

Figure 3B. Variable cross-reference list.

FNT	5	90	110	130	3010	3010	3010				
X	5	5	540	541	2060	2160					
LI[ ]	30	270	2160	4030							
SI[ ]	30	130	3030								
RI[ ]	30	220	670	680	6020	6030					
AI[ ]	30	190	830	4080	4080	6040	6040				
WI[ ]	30	350	360	370	510	510	510	2060	2140	2180	3010
	3010	3010	4060								
I	75	130	140	140	200	220	240	250	270	290	
L	80	90	100	170							
L1	90	130									
M	100	110	120	160	6030	6040	6040				
M1	110	130									
R	120	130	150								
J	210	220	230	260	270	280					
B	340	520	690	710	2110	4030					
I0	380	615	670	680	700	700	6010				
Y	540	541	2110	2140	2160	2180					
L2	570	2030	2190								
M4	640	660	680	690	4010	4075	4080	4100			
S3	670	3020	3030	3040	4080	4080					
W	740	770	6040								

Z	810	820					
I2	2040	2050	2060	2070	2130	2140	2150
J2	2050	2180					
A3	3010	3030					
I4	4020	4030	4060	4080	4100	4110	
J4	4050	4060	4070				
I6	6010	6020	6030	6050			
S	6020	6040	6040				

Another new game from Creative Computing . . .

Marooned In Space TRS-80 Action Game

William L. Colsher

In orbit around an unknown planet, your ship's navigational computer disabled, you are unable to plot a course to return to Earth until the planet is identified. Fortunately, the survey sensors aboard ship still function and they can supply you with information on the physical characteristics of the planet. From this information, you must determine which planet you orbit. If you are unable to identify the planet after all the information has been supplied, the computer, having eliminated wrong guesses, will identify the planet and then be able to get you home.

I came up with the idea for this interesting game while reading The Promise of Space by Arthur C. Clarke. In fact, all the data on the planets was obtained from that book. Just reading a table of information has always been a difficult way for me to learn anything so I wrote this game to make it a little more fun.

In addition to being fun to play, even for someone with no prior knowledge of the solar system, this game also makes interesting use of a couple of the features of the Level-I TRS-80. Graphics are used in a rather limited manner to provide separation for various parts of the display. More interesting is the use of the "PRINT AT" statement. By dividing the number to be displayed by 100 (you can use any integral power of 10 and be safe) and then placing the "PRINT AT" statement in a "FOR-NEXT" loop indexed from 1 to 100 you can get a very interesting, dynamic display. The general consensus of opinion from those who've played the game is that using this technique adds to the game simply because there seems to be so much more going on. It makes the display "feel" more like an actual read-out of some instrument. □

William L. Colsher, 4328 Nutmeg Lane, Apt. 111, Lisle, IL 60532.

```
10    REM***MAROONED IN SPACE
20    RESTORE
22    O=O:Z=O
30    CLS:P.A.463,"M A R O O N E D  I N  S P A C E"
40    F.I=1TO500:N.I
50    CLS:IN."DO YOU NEED INSTRUCTIONS (Y=1, N=2)";A
60    IF A=1 T. GOS. 1000
70    CLS:IN."WILL YOU USE THE METRIC (1) OR ENGLISH (2) SYSTEM";A
72    B=1:IF A=1 THEN B=1.61
80    CLS:F.X=92TO127:S.(X,1):S.(X,35):N.X
90    F.Y=1TO35:S.(92,Y):S.(93,Y):S.(126,Y):S.(127,Y):N.Y
100   P.A.115,"PLANETS";
110   P.A.175," 1. MERCURY";:P.A.239," 2. VENUS";:P.A.303," 3. MARS";
120   P.A.367," 4. CERES";:P.A.431," 5. PALLAS";:P.A.495," 6. JUPITER";
130   P.A.559," 7. SATURN";:P.A.623," 8. URANUS";:P.A.687," 9. NEPTUNE";
160   P.A.849,"ASSOCIATIVE LOGIC UNIT DAMAGED";
165   P.A.901,"HUMAN INTERVENTION REQUIRED. ORBIT STABLE OVER UNKNOWN"
167   P.A.967,"PLANET. PLANETARY SURVEY DATA IS IN PREPARATION.";
170   F.I=1TO25:P.A.795,"              ";:F.J=1TO100:N.J:P.A.795,"EMERGENCY";
180   F.J=1TO100:N.J:N.I
190   P.A.768," ":P.A.832," ":P.A.896," "
200   P.A.967,"                                                  ";
220   P=RND(9)
222   IF P=0 THEN 220
224   O=P
230   F.I=1TOP:READA(1),A(2),A(3),A(4),A(5),A(6),A(7):N.I
240   P.A.64,"PLANETARY SURVEY DATA";
310   M=A(1)/100
320   P.A.192,"MASS";
330   F.I=1TO100:P.A.212,M*I;:N.I
```

Marooned, con't...

```
340     P."(EARTH=1)";
350     GOS.2000
410     G=A(2)/100
420     P.A.256,"SURFACE GRAVITY";
430     F.I=1TO100:P.A.276,G*I;:N.I
440     P."(EARTH=1)";
450     GOS.2000
510     V=A(3)*B/100
520     P.A.320,"ESCAPE VELOCITY";
530     F.I=1TO100:P.A.340,V*I;:N.I
540     IFB=1.61T.P."K.P.S.";
542     IFB=1T.P."M.P.S.";
550     GOS.2000
610     D=A(4)*B/100
620     P.A.384,"DIAMETER";
630     F.I=1TO100:P.A.404,D*I;:N.I
640     IFB=1T.P."MILES";
642     IFB=1.61T.P."KILOMETERS";
650     GOS.2000
710     P.A.448,"NUMBER MOONS";
720     F.I=0TOA(5):P.A.468,I;:N.I
750     GOS.2000
810     Y=A(6)/100
820     P.A.512,"LENGTH OF YEAR";
830     F.I=1TO100:P.A.532,Y*I;:N.I
840     P."(EARTH=1)";
850     GOS.2000
910     R=A(7)*B*10000
920     P.A.576,"DISTANCE FROM SUN";
930     F.I=1TO100:P.A.596,R*I;:N.I
940     IFB=1T.P."MILES";
942     IBB=1.61T.P."KILOMETERS";
950     GOS.2000
955     IFZ=1T.3000
960     P.A.768,"I NOW HAVE ALL NECESSARY INFORMATION. THE PLANET WE"
965     P.A.832,"ARE ORBITING IS ";
967     ONOG.991,992,993,994,995,996,997,998,999
970     P."I CAN NOW PLOT A COURSE FOR HOME.";
975     G.3000
991     P."MERCURY";:G.970
992     P."VENUS";:G.970
993     P."MARS";:G.970
994     P."CERES";:G.970
995     P."PALLAS";:G.970
996     P."JUPITER";:G.970
997     P."SATURN";:G.970
998     P."URANUS";:G.970
999     P."NEPTUNE";:G.970
1000    CLS:P.A.15,"MAROONED IN SPACE";
1010    P.A.128,"IN THIS GAME YOU ARE MAROONED IN ORBIT AROUND AN UNKNOWN"
1020    P.A.192,"SOLAR PLANET. A MALFUNCTION HAS PREVENTED YOUR COMPUTER"
1030    P.A.256,"FROM IDENTIFYING IT. THE COMPUTER CAN PROVIDE YOU WITH"
1040    P.A.320,"INFORMATION ON THE PLANET FROM WHICH YOU MUST FIGURE"
1050    P.A.384,"OUT WHICH PLANET IT IS."
1060    P.A.512,"WHE THE PLANET IS IDENTIFIED, THE COMPUTER CAN PLOT A"
1070    P.A.576,"COURSE FOR HOME. "
1075    P.A.960,"(ENTER TO PLAY)";:IN.A$
1080    RET.
2000    REM***ASK WHAT PLANET
2005    IFZ=1THEN RET.
2010    P.A.768,"WHAT PLANET ARE WE PRESENTLY ORBITING";:IN.A
2015    IFA=0THEN2500
2020    P.A.768,"DATA NOT CONSISTENT WITH INFORMATION ON THAT PLANET"
2030    P.A.832,"MORE INFORMATION IS IN PREPARATION."
2040    F.I=1TO600:N.I:P.A.768," ":P.A.832," ":RET.
2500    P.A.768,"MY DATA INDICATE THAT YOU ARE CORRECT. A COURSE FOR"
2510    P.A.832,"EARTH HAS BEEN PLOTTED. WE'LL BE HOME SOON."
2520    P.A.896,"FOR YOUR INFORMATION, HERE IS THE REST OF THE DATA ON";
2530    P.A.960,"THE PREVIOUSLY UNKNOWN PLANET.";
2540    Z=1:RET.
3000    P.A.960,"WILL YOU PLAY AGAIN (Y=1, N=2)";:IN.A
3010    IF A=1 THEN 10
3020    STOP
10000   D..05,.37,2.5,3000,0,.24,36
10010   D..81,.89,6.2,7600,0,.61,67
10020   D..11,.39,3.1,4200,2,1.89,142
10030   D..0002,.1,.4,480,0,4.6,256
10040   D..00005,.05,.2,300,0,4.6,257
10050   D.318,2.65,38,86000,12,11.86,484
10060   D.95,1.15,23,74000,10,29.5,895
10070   D.15,1,14,30000,5,84,1780
10080   D.17,1.55,15,28000,2,164,8,2790
```

Thinkers' Corner

© Layman E. Allen

WORD PUZZLES

How many of the problems (a) through (f) below can you solve by forming a network of words that have exactly as many letters as the number listed as the GOAL? (Suppose that each symbol below is imprinted on a disc.)

To qualify as a network

(1) all sequences of discs across and down must be words,
(2) the words must have two or more letters and not be proper names,
(3) all of the discs in the REQUIRED column must be used,
(4) as many of the discs in PERMITTED as you wish may be used, and
(5) at most one of the discs in RESOURCES may be used.

Example: The number of letters in the words of the network

```
CAT     is 7:          CAT=3, TO=2, ON=2
 ON                     3 + 2 + 2 = 7
```

The number in the network CAT is 3.

PROB.	GOAL	REQUIRED	PERMITTED	RESOURCES
[a]	6	N	E O T	C E G M O R S
[b]	8	A B	B O R Y	A D F M N O R
[c]	6	H W	A E S	A B E F N O R
[d]	8	A B E	F M T	B D F I N V W
[e]	9	F Y	A N O U	C F M O Q R U
[f]	10	A N	O R T	B D G M N O S

Some Suggested Answers (frequently there are others):

```
[a] O          [b] B A B Y     [c]   S
    N O T          O               W H O
    E              R                 E
                   N

[d]   N        [e] M A N Y     [f] R A N
      A T                O F         T O O
      M                  U
    B E
```

If you enjoy this kind of puzzle, you may like playing ON-WORDS: The Game of Word Structures. Free information about this and other instructional games is available upon request from The Foundation for the Enhancement of Human Intelligence, 1900-W Packard Road, Ann Arbor, MI 48104.

A 6502 Disassembler - In Microsoft BASIC

Anthony T. Scarpelli

The following is a program that will disassemble the 6502 op-codes. It will print out the address, the op-code, any data, all in hex, and the mnemonics and addressing mode. The program is written in Microsoft BASIC. It's a program that looks more complicated than it is, but if you follow the flow charts and the program itself you'll see that it is arranged into a system that could be adapted for other microprocessors (e.g., the Apple and PET).

My impetus for writing this program came from many directions. The first was that I wanted to get into a program that had no listing. Secondly, a disassembler program, written by Jef Raskin for the 8080 and published in the April '79 edition of **73**, made me confident enough that it was possible to write one in BASIC; and, thirdly, Jim Butterfield put all the op-codes into some semblance of logical order, and can be seen in issue #13 of the **6502 User Notes**.

If you have ever programmed in assembly language, you know that the op-codes are given names which are called mnemonics. For example, LoaD Accumulator becomes LDA. Also, op-codes use different amounts of memory space, either 1, 2 or 3 bytes. Therefore, the problem is two-fold. We have to be able to obtain from the op-code the amount of memory used as well as a means to point to a table of mnemonics. Because there was a method to the MOS Technology madness in the construction of the 6502 op-codes, a way of breaking up the codes into a right and left string was used and using those strings to get the above information.

The first thing we do is initialize all the necessary arrays, data, strings and one of two flags from lines 1 to 98. As you can see, all the mnemonics are put into a string table from 31 to 90 and the addressing mode table is from 91 to 98. By the use of masks on the data these strings will be pointed to.

Anthony T. Scarpelli, RR#1, Box 426, North Windham, ME 04062.

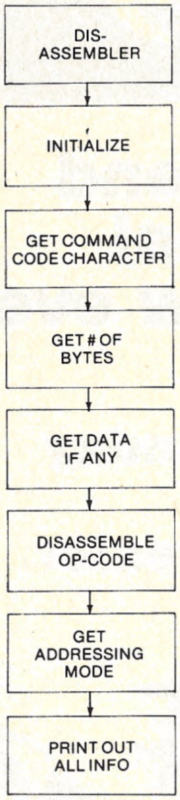


Figure 1. Disassembler program overall flowchart.

The main program is from 100 to 105. It is a little weird to have a 5-line main program but I did it this way so that everything else is subroutine. I have two command codes only because of my limited memory space. Yet if you want, simply by adding more codes you can have a more versatile program. It is a simple matter to add command codes to go forward one op-code at a time, to go back one op-code, to branch, to jump to subroutines and return from one, and also to search the whole disassembled program for a particular op-code and/or data. The command code "A" will ask for an address, and "/" will start the disassembly process from that address. In line 101, the GET C$ is Microsoft's way of inputting a single character string and going to the next line without hitting return. It's good in some ways, but the only way of stopping the program is to hit the RETURN after the normal INPUT in the "A" Start Address subroutine. "?" is a print statement and ";" allows multiple statements in a line.

In this first of the subroutines you input your address in hex, then the string is converted to decimal by a handy little subroutine, because BASIC uses decimal in all its computations and I didn't want to go through the calculator hastle. We return back to the command code mode.

The ' "/" Continue 17 Lines' subroutine sets up a loop that will display the number of lines you want. It also goes to the meat of the program as a subroutine, and tests the X flag which is used in case the op-code that was disassembled is not an op-code, which will happen in case you run into tables and other non op-code things. By the repeated use of the "/", which is near the RETURN key, you disassemle the whole program. By the way, if you don't have a printer, and want hard copy of the program you are taking apart, you can take pictures of the screen using Tri-X film with a shutter speed of 1/30th second and a lens opening of f/5.6.

The first thing we do at 150 is get the op-code at the start address, change the decimal back to a hex string and pick off the right half and

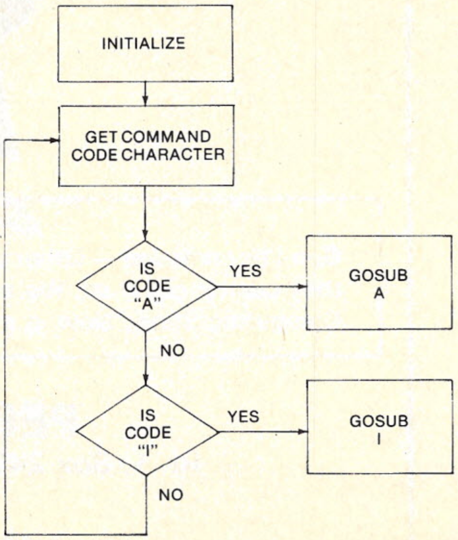
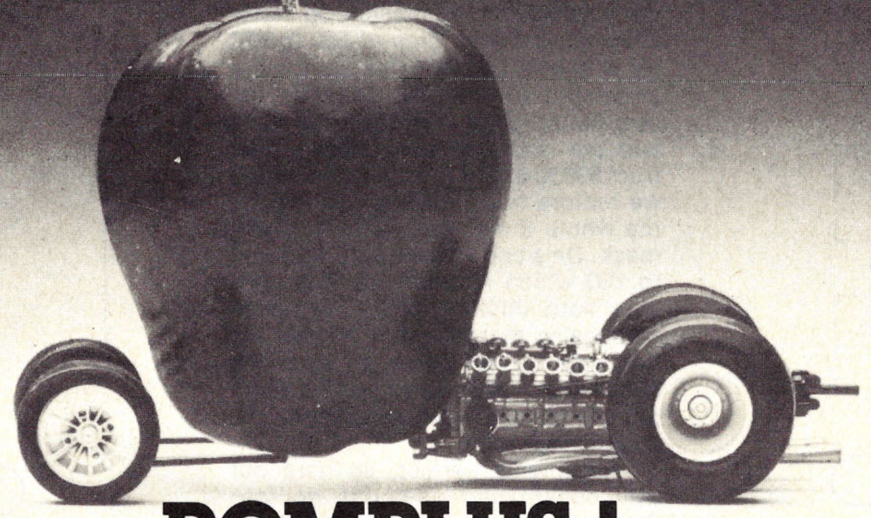


Figure 2. Main program.

NEW POWER FOR YOUR APPLE FROM MOUNTAIN HARDWARE.



ROMPLUS+

NEW EXPANDABILITY.
ROMPLUS+ is a peripheral board whose added features can turn the Apple * computer into the most powerful personal computer available today.

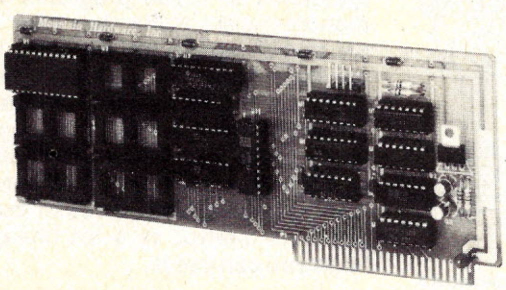
NEW POWER.
ROMPLUS+ provides six sockets to accept individually addressable 2K ROM's or EPROM's. Keyboard Filter™, a 2K ROM program, comes installed on the ROMPLUS+ board and adds many useful features to your Apple, including:

- Upper and lower case letters. The only system that offers keyboard input and standard shift key operation.
- Multiple user-defined character sets.
- Colored or inverse-colored letters.
- Keyboard macros—two key-stroke, automatic typing of multiple, user-defined words or phrases. Including BASIC and DOS commands.
- Mixed text and graphics.
- Improved cursor control.
- STOP LIST and END LIST.
- Works with Integer BASIC, RAM or ROM Applesoft, and DOS.
- And more...quick to learn. Easy to use.
- Software support provided on disk includes demonstration programs and two Editors that allow you to define your own characters or keyboard macros.

SOPHISTICATED FIRMWARE.
In addition to the Keyboard Filter ROM, ROMPLUS+ offers five sockets for ROM or EPROM, plus "scratch-pad" RAM. And, sophisticated firmware on ROMPLUS+ allows one, two, or more of its chips to be used simultaneously for programs longer than 2K.

EXPANDED UTILITY.
Many software programs really ought to be utilized as firmware. ROMPLUS+ makes that an actuality for the Apple by providing six additional ROM sockets.

AT YOUR DEALER. NOW.
$169. Complete and tested. Including the powerful Keyboard Filter ROM and full documentation. Ask your dealer for a demonstration.



Mountain Hardware, Inc.
LEADERSHIP IN COMPUTER PERIPHERALS
300 Harvey West Blvd., Santa Cruz, CA 95060
(408) 429-8600

Sounds great.
☐ Please send me all the details on ROMPLUS+ and Keyboard Filter.

Name ____________________

Address ____________________

City ____________ State ________ Zip ______

Phone ____________________

*Trademark of Apple Computer Inc.

Disassembler, con't...

left half of the hex number. The clue to whether the op-code is a 1, 2, or 3 byte instruction is contained in the hex number, and as you can see by the flow chart, it's fairly easy to dig this information out. Next, because our final printout is going to be a

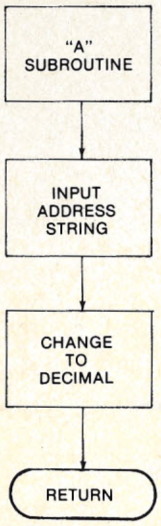


Figure 3. Address subroutine.

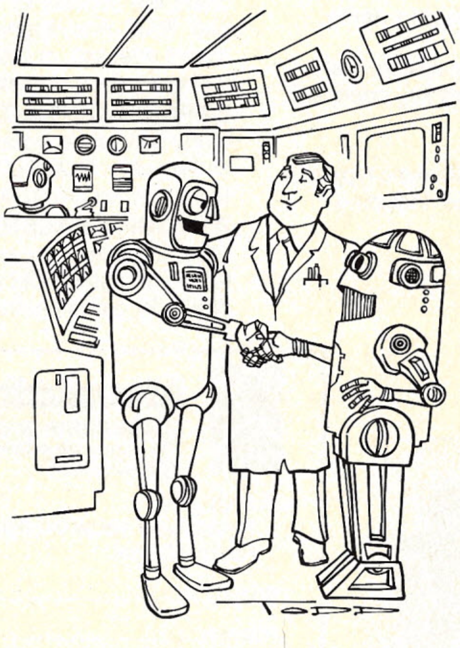
string, we'll change the address to a string at 340, and then get our spaces in our final string set so that everything is all lined up. If the instruction is one byte long we go directly to the disassemble subroutine. Otherwise we have to collect any data after the op-code and put them into a string.

After we have the mnemonics and address mode at 450 we simply print out all the information in 460, then we have to get all spaces and data strings down to nothing so no data is reprinted, and increment the address by one so we can disassemble the next op-code. Just in case there was no op-code decodable, we test a flag and skip the printing.

The 'Disassemble Op-Code' subroutine is a little more complicated, but if we want mnemonics we have to go through it. We go back to our broken hex number and it tells us what we have to do. All the branch and other instructions that have implied addressing mode have been grouped together. Their hex numbers were converted to decimal and are located in the first 38 data numbers in lines 5, 6, 7 and 8. In lines 660 to 690 we have a loop that reads the data and compares it to the decimal number of the op-code. Once there's a match we can pick off the counter and use it to point to the mnemonic as in 710 right after we restore our data in 700.

In case the op-code is not an implied instruction we have to do a different route. We first set a flag to indicate that this other direction was taken. Next we set a mask loop. Three bits of the binary numbers which makes up the op-code indicates the addressing mode. If you mask out these bits, the resulting number is the same for a particular instruction no matter what the addressing mode is. This masked number can then be used in the same way as above in a counting loop. Seven different masks are needed and the mask loop starts at statement 540. Then at 550 to 570 we skip over the data that we are not interested in. At 580 we AND the decimal op-code with the first mask and our compare loop comes next from 590 to 620. If there was no match we restore our data and go through the whole thing again with our next mask. On a correct compare we jump to 700 which restores the data and here both implied and non-implied addressing merge again. In case the flag was not set we return from the subroutine. Otherwise we still have to get the string which represents the addressing mode.

Nothing is perfect, so here our well constructed op-codes go into that land of otherwise. "R" is our pointer in the addressing mode strings from 91 to 98. From 722 to 729 we take care of all those oddball op-codes. At 730 we AND the decimal op-code with a mask and after dividing by four we have our pointer



"Hello, I'm 912YZ, but everyone usually calls me by my nicknumber, 912WX..."

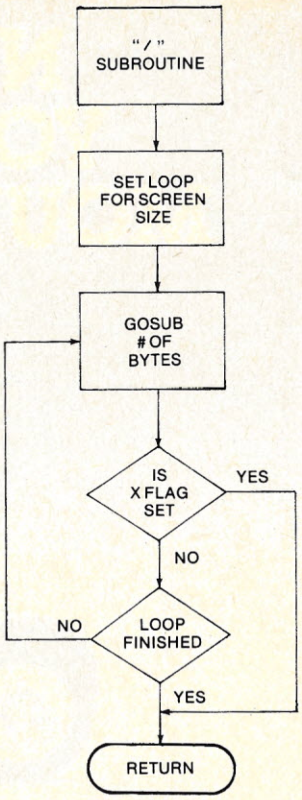


Figure 4. Continue subroutine.

which helps to get our string at 740. After clearing our flag, back we go to start the whole process over again at the next address.

In case the data retrieved at a particular address wasn't an op-code, both search loops fall through to a "no op-code" message after a flag is set. This way we can jump back through all the returns to the command code so we can get a new address. Another possibility here is that because the data is probably a table you can put in a subroutine that will merely display the data until you find a definite op-code.

There are two other subroutines which I won't go into. They merely change the decimal numbers to hex and vice versa. They are common enough for no explanation.

Fun, wasn't it? I hope you can use this program to delve into the mysteries of those programs which manufacturers think they can close off to our prying little minds. When we have kludged together systems and want them to work with available software that provides no listing, this is the only method we have available to us to get the needed information without paying thousands of dollars. And this method is a lot cheaper. I hope you can use it. □

References.

"6502 Op Codes," Jim Butterfield. **6502 User Notes**, No. 13. "An 8080 Disassembler," Jef Raskin. **73 Magazine**, April, 1979.

Disassembler, con't...

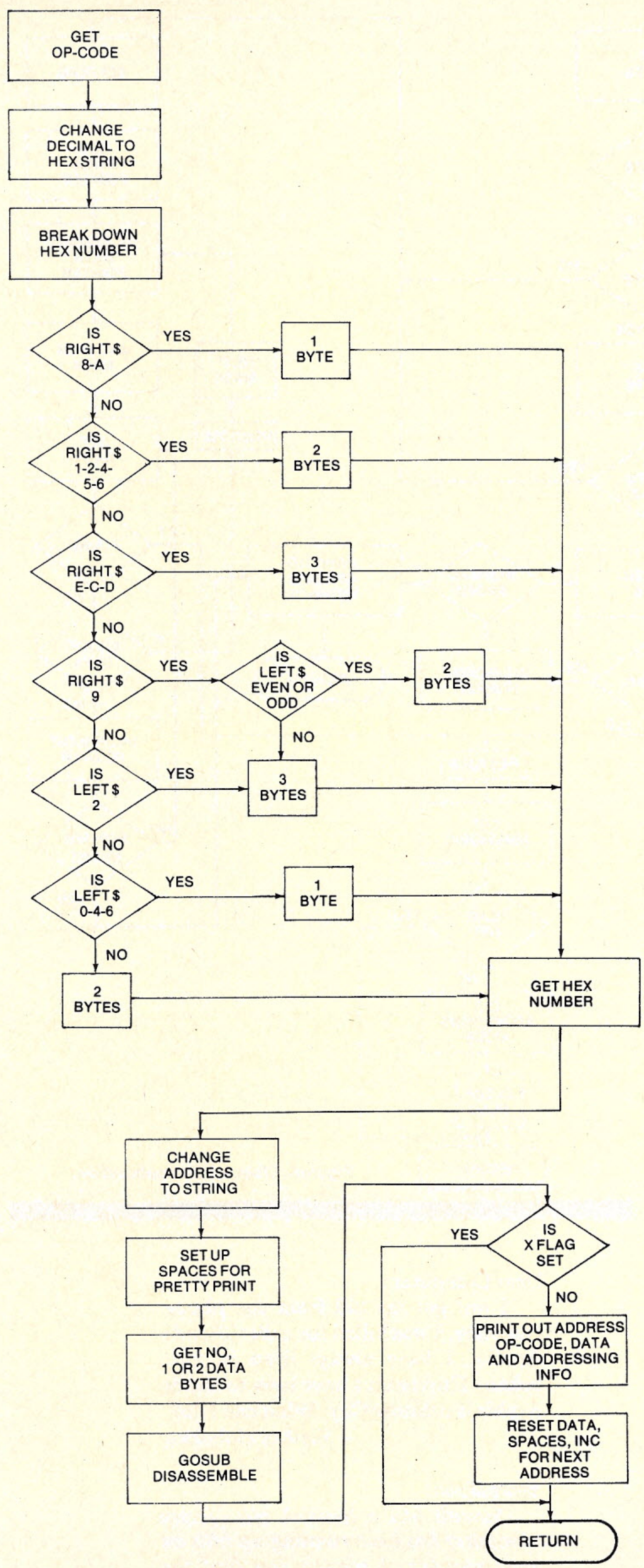


Figure 5. Get # of bytes subroutine.

Disassembler, con't...

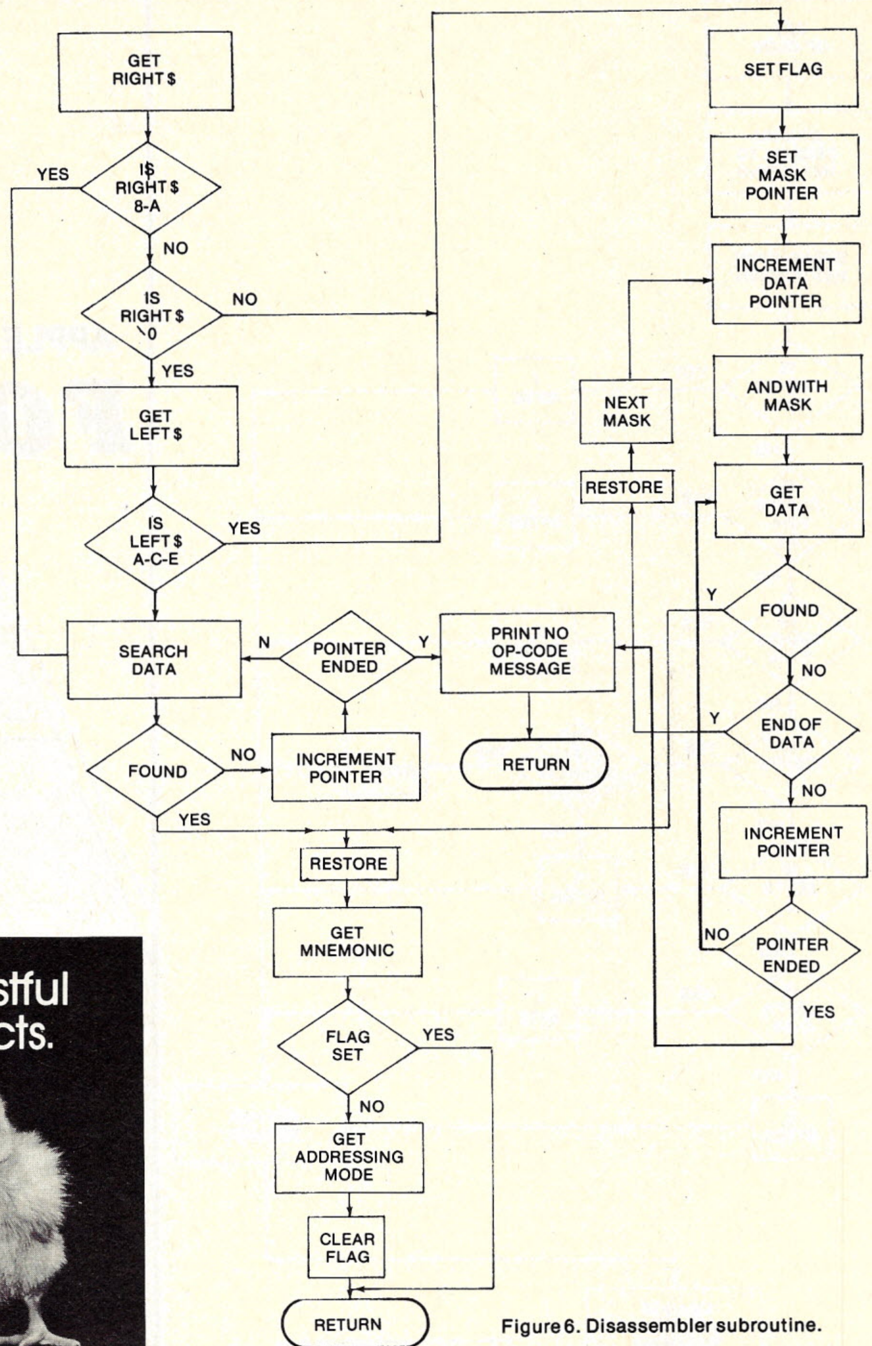


Figure 6. Disassembler subroutine.

Dear Computer:

I just got my bill from the phone company. I was billed for calls to Saudi Arabia. I have always lived in Hoboken. The closest I have been to Saudi Arabia is Atlantic City. What can I do?

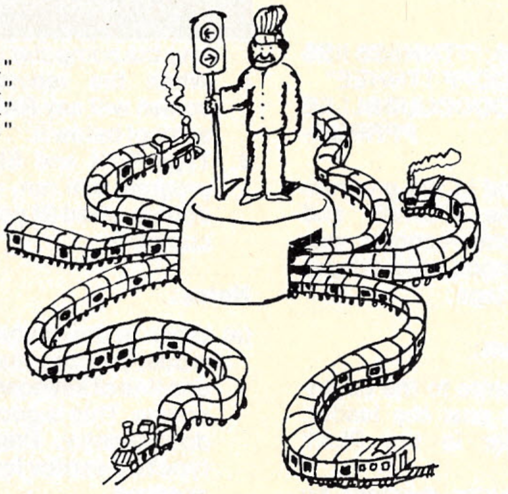
— Sheik of Hoboken

Dear Hobo:

Sounds like a lovesick New Jersey computer has been running up bills on your number. I suggest you find the offending machine and fold, spindle, and mutilate it.

Disassembler, con't...

```
0 REM 6502 DISASSEMBLER
1 DIM M(7)
2 DIM S(4)
3 DIM P$(60)
4 DIM S$(7)
5 DATA 8,24,40,56,72,88,104,120,136,152,168,184,200
6 DATA 216,232,248,10,42,74,106,138,154,170,186,202
7 DATA 234,0,64,96,16,48,80,112,144,176,208,240
8 DATA 32,2,34,66,98,134,162,198,230,36,132,160
9 DATA 192,224,1,33,65,97,129,161,193,225,76
11 M(1)=227
12 M(2)=195
13 M(3)=231
14 M(4)=247
15 M(5)=243
16 M(6)=223
17 M(7)=28
18 S(1)=4096
19 S(2)=256
20 S(3)=16
21 S(4)=1
22 H$="0123456789ABCDEF"
23 F=0
31 P$(1)="PHP"
32 P$(2)="CLC"
33 P$(3)="PLP"
34 P$(4)="SEC"
35 P$(5)="PHA"
36 P$(6)="CLI"
37 P$(7)="PLA"
38 P$(8)="SEI"
39 P$(9)="DEY"
40 P$(10)="TYA"
41 P$(11)="TAY"
42 P$(12)="CLV"
43 P$(13)="INY"
44 P$(14)="CLD"
45 P$(15)="INX"
46 P$(16)="SED"
47 P$(17)="ASL-A"
48 P$(18)="ROL-A"
49 P$(19)="LSR-A"
50 P$(20)="ROR-A"
51 P$(21)="TXA"
52 P$(22)="TXS"
53 P$(23)="TAX"
54 P$(24)="TSX"
55 P$(25)="DEX"
56 P$(26)="NOP"
57 P$(27)="BRK"
58 P$(28)="RTI"
59 P$(29)="RTS"
60 P$(30)="BPL"
61 P$(31)="BMI"
62 P$(32)="BVC"
63 P$(33)="BVS"
64 P$(34)="BCC"
65 P$(35)="BCS"
66 P$(36)="BNE"
67 P$(37)="BEQ"
68 P$(38)="JSR"
69 P$(39)="ASL"
70 P$(40)="ROL"
71 P$(41)="LSR"
72 P$(42)="ROR"
73 P$(43)="STX"
74 P$(44)="LDX"
75 P$(45)="DEC"
76 P$(46)="INC"
77 P$(47)="BIT"
78 P$(48)="STY"
79 P$(49)="LDY"
80 P$(50)="CPY"
81 P$(51)="CPX"
82 P$(52)="ORA"
83 P$(53)="AND"
84 P$(54)="EOR"
85 P$(55)="ADC"
86 P$(56)="STA"
87 P$(57)="LDA"
88 P$(58)="CMP"
89 P$(59)="SBC"
90 P$(60)="JMP"
91 D$(0)="(I,X)"
92 D$(1)="Z"
93 D$(2)="I"
94 D$(3)="A"
```



```
95 D$(4)="(I),Y"
96 D$(5)="Z,X"
97 D$(6)="A,Y"
98 D$(7)="A,X"
99 REM MAIN PROGRAM
100 X=0
101 ?"COMMAND CODE ";:GET C$
102 ?:IF C$="A" THEN GOSUB 113
103 ?:IF C$="/" THEN GOSUB 118
105 GOTO 100
112 REM "A" START ADDRESS
113 INPUT "ADDRESS";A$:N$=A$
114 GOSUB 1000
115 A=D
116 RETURN
117 REM "/" CONTINUE 17 LINES
118 FOR Q=1 TO 17
119 GOSUB 150
120 IF X=1 THEN 122
121 NEXT Q
122 RETURN
149 REM GET # OF BYTES
150 OP=PEEK(A):N=OP
160 GOSUB 1200
170 OP$=X$
180 R$=RIGHT$(OP$,1):L$=MID$(OP$,3,1)
190 IF R$="8" OR R$="A" THEN B=1:GOTO 310
200 IF R$="1" OR R$="2" OR R$="4" THEN B=2:GOTO 310
210 IF R$="5" OR R$="6" THEN B=2:GOTO 310
220 IF R$="E" OR R$="C" OR R$="D" THEN B=3:GOTO 310
230 IF R$="9" THEN 270
240 IF L$="2" THEN B=3:GOTO 310
250 IF L$="0" OR L$="4" OR L$="6" THEN B=1:GOTO 310
260 B=2:GOTO 310
270 N$="000"+L$
280 GOSUB 1000
290 IF D=INT(D/2)*2 OR D=0 THEN B=2:GOTO 310
300 B=3
310 OP$=RIGHT$(OP$,2)
320 N=A
330 GOSUB 1200
340 A$=X$
350 IF B=3 THEN S$=" ":S2$="   ":GOTO 390
360 IF B=2 THEN S$="":S2$="      ":GOTO 420
370 S2$="        "
380 GOTO 450
390 A=A+1:N=PEEK(A)
400 GOSUB 1200
410 D2$=RIGHT$(X$,2)
420 A=A+1:N=PEEK(A)
430 GOSUB 1200
440 D1$=RIGHT$(X$,2)
450 GOSUB 500
455 IF X=1 THEN 480
460 ?A$;"  ";OP$;S$;D2$;" ";D1$;S2$;NM$;G$
470 D1$="":D2$="":S$="":S2$="":G$="":A=A+1
480 RETURN
499 REM DISASSEMBLE OPCODE
500 IF R$="8" OR R$="A" THEN 660
510 IF R$<>"0" THEN 530
520 IF L$="A" OR L$="C" OR L$="E" THEN 530
525 GOTO 660
530 F=1
540 FOR J=1 TO 7
550 FOR I=1 TO 38
560 READ M
570 NEXT I
580 C=OP AND M(J)
590 FOR I=39 TO 60
600 READ M
610 IF C=M THEN 700
620 NEXT I
630 RESTORE
640 NEXT J
650 ?"NO OPCODE AT NEXT ADDRESS"
651 X=1
652 ?"NEXT ";:RETURN
660 FOR I=1 TO 38
670 READ M
680 IF M=OP THEN 700
690 NEXT I
695 GOTO 650
700 RESTORE
710 NM$=P$(I)
720 IF F=0 THEN 760
722 IF OP$="A0" OR OP$="C0" OR OP$="E0" THEN R=2:GOTO 740
724 IF OP$="A2" THEN R=2:GOTO 740
726 IF OP$="BE" THEN R=6:GOTO 740
728 IF OP$="96" OR OP$="B6" THEN G$=",Z,Y":GOTO 750
729 IF OP$="6C" THEN G$=",IND":GOTO 750
730 R=(OP AND 28)/4
740 G$=","+D$(R)
750 F=0
760 RETURN
999 REM HEX TO DECIMAL ENTER WITH N$ EXITS WITH D
1000 J=1
1010 FOR I=1 TO 4
1020 D(I)=0
1030 NEXT I
1040 FOR I=1 TO 4
1050 D(I)=ASC(MID$(N$,J,1))-48
1060 IF D(I)>9 THEN D(I)=D(I)-7
1070 J=J+1
1080 NEXT I
1090 D=4096*D(1)+256*D(2)+16*D(3)+D(4)
1100 RETURN
1199 REM DECIMAL TO HEX ENTER WITH N EXITS WITH X$
1200 X$=""
1220 FOR J=1 TO 4
1225 P=S(J)
1230 FOR I=1 TO 16
1240 IF N-I*P<0 THEN 1260
1250 NEXT I
1260 X$=X$+MID$(H$,I,1):N=N-(I-1)*P
1300 NEXT J
1310 RETURN
```

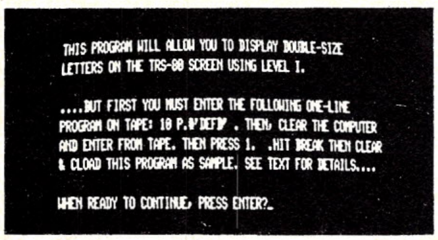
TRS-80

Double-Size Characters

Fred Blechman

If you have a BASIC Level I TRS-80, you may want to display double-size characters on your screen - but you've been told that this is only possible with Level II installed. Not so! By "tricking" the BASIC interpreter, you can access the character generator in the TRS-80, which is able to display double-size characters. Instead of 16 lines of 64 characters each, you can display 16 lines of 32 double-sized characters per line.

You can "confuse" the TRS-80 if you don't confuse yourself first. Follow these instructions step-by-step, including the example. After you've done it a few times, you'll find it very easy to do — much easier to do than to explain! Don't try to UNDERSTAND it - just do it!

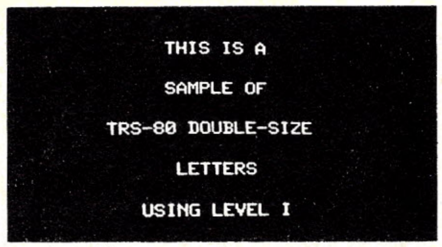


Generating The Program:

1. Turn on the computer and check the memory by typing in P.M. and ENTER. The display should read 3583 or 15871.
2. Type in this one-line program and ENTER:
   10 P.#"DEFB"
3. Put a blank cassette in your recorder (make sure it's advanced past the leader) and set the recorder in RECORD.
4. Type RUN and ENTER. The tape recorder will run briefly and record the one-line program. Leave the recorder in RECORD.
5. Type in the desired program in the regular way, making sure to use line 10. ALL PRINT and PRINTAT statements must use repeated characters and spaces, since this arrangement ignores every other character or space. Try this example:
   5 CLS
   10 PRINTAT 390;"TTHHIISS IISS AA TTEESSTT OOFF TTHHEE"
   20 PRINT "DDOOUUBBLLEE LLEETTEERR PPRROOGGRRAAMM"
6. Once you've typed in the program you want, type CSAVE and ENTER. Since the recorder was left in RECORD mode, this will put your program on tape.

Running The Program:

A. Rewind the cassette to the beginning, advancing past the leader. Put the recorder in the PLAY mode.
B. Clear the computer memory by typing in NEW and ENTER. P.M. should read 3583 or 15871.
C. Type CLOAD and ENTER. The tape will run very briefly and stop. The screen will say READY.
D. STOP the recorder, or the next operation will cause it to run.
E. Type 1. and ENTER. Do not forget to type a period after the 1. The display will shift to the right and then, after a brief pause, will display double-size letters. But only every other character is displayed. For example, instead of READY you'll see RAY. Notice that the prompt and cursor are also twice-size!
F. Press BREAK and then CLEAR keys.
G. Now put recorder in PLAY again.
H. Type CLOAD and ENTER. The recorder will load your program



into the computer memory.

I. When the recorder stops, the screen will say RAY with a prompt symbol below it.
J. Type RUN and ENTER and your program will run, with all the text double-sized (if your program was correct!)

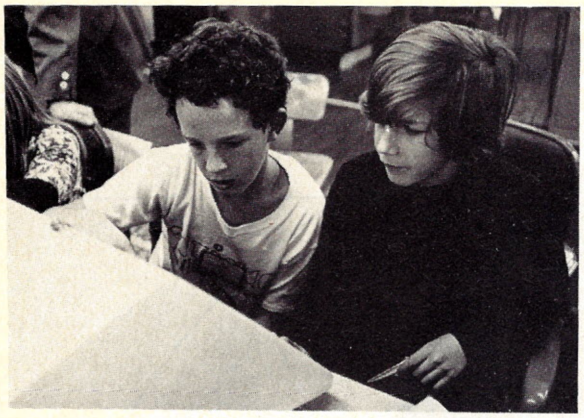
Notes:

(a) When in the double-size character mode, you can make slightly over one line of keyboard entries on the screen. This is good for practice in double-entry. Press BREAK and type two entries for each character or space.

(b) Exiting the program completely can only be done by SHUTTING OFF the computer! When you press the RESET switch at the left rear of the keyboard (under the flip-up panel), you return to BASIC, and you can run or list your program, but you can't CHANGE the program! The computer, in its confused state, shows up with a NEGATIVE memory when P.M. is entered, so you only get a SORRY if you try to change instructions. Repeated pressing of the RESET switch will not get you out of the program - only shutting off the computer will do it. Leave it off for 5 or 10 seconds, turn it back on and check memory with P.M.

To repeat, this all sounds much tougher than it is. Try it - you'll like it!

Fred Blechman, 7217 Bernadine, Canoga Park, CA 91307.



Micros "GOTO" School

Donald T. Piele

Microcomputers can be used in the classroom for: instructional activities that we associate with CAI (Computer Assisted Instruction) or CMI (Computer Managed Instruction); enrichment activities that we associate with simulations and games; making numerical calculations for the purpose of solving mathematical problems; teaching students how to program - primarily in the BASIC language; and individual exploration of original problem-solving.

Each lesson consists of a simple program with a short explanation of the new statements, a sample run, and a series of simple program changes for the student to do.

This article is focused on the latter activity. It is a report of a pilot project in which a microcomputer was placed in a sixth grade classroom for 8 weeks for the purpose of developing logical thinking skills. The students were given instruction on how to program the APPLE II microcomputer to draw color graphics designs. They were then given similar problems to solve using the commands they had learned.

An Apple For The Teacher

In the Spring of 1978, I contacted Gordon Kunaschk, a sixth grade teacher at Bose Elementary School in Kenosha, Wisconsin. He was receptive to the idea of giving up two hours a week of class time for eight weeks to let me teach his sixth graders how to program a microcomputer. If nothing else, it would be a lesson in computer literacy. Gordie had never programmed a computer before, but he was willing to learn along with the kids if I was willing to provide a computer and the necessary instruction.

The Center For The Application of Computers at UW-Parkside supported the idea and supplied an APPLE II microcomputer for the project. This was a fortunate choice for us since the APPLE II system is easy to use: it is portable; it has a good keyboard; and most important of all, it has a very simple and natural set of graphics commands that allow the programmer to create pictures on a TV screen using 16 different colors. The ideas I wanted to emphasize about computer programming would be considerably enhanced by a graphics display. The basic programming construct of a loop, for example, could be visualized, and every problem to be solved by the students could be represented by a single picture.

Getting Started

My objective for bringing a microcomputer into a sixth grade classroom was to create an environment for **active problem-solving**. The BASIC programming language statements, enhanced by the graphics of the APPLE II microcomputer, form the logical building blocks. Each lesson consists of a simple program with a short explanation of the new statements, a sample run, and a series of simple program changes for the student to do. These activities allow the student to discover how the statements in the program effect its outcome. Also, problems are posed that require the student to combine statements in sequential order to solve a problem. As a result of working on these questions, the student gets a working understanding of practical problem-solving skills such as:

1. Understand the problem, its givens and goals
2. Make conjectures and probe the problem by trial and error
3. Decide on a set of possible methods of attack
4. Evaluate each possible approach for its correctness
5. Reflect on successful solutions and generalize

Each of the following exercises was designed to provide practice for these skills. They are samples taken from a larger collection and are not contiguous lessons.

Lesson #1

Key Words: GR, COLOR, PLOT

LIST	EXPLANATION
10 GR 20 COLOR = 9	The computer is put in GRaphics mode.
20 COLOR = &	The COLOR is set to orange. There are 16 different colors to choose from.
30 PLOT 10,15	The position 10 over, 15 down from the upper left hand corner is plotted.
100 END	

RUN

Programming the microcomputer was considered by the sixth graders to be highly motivating. They would rather spend their recess on it than go outdoors.

Your Turn (RUN the program after each change)

1. Change line 20 to . . 20 COLOR = 3
2. Change line 30 to . . . 30 PLOT 5,7
3. Add line 50 . . . . 50 PLOT 5,5
4. Add line 40 . . . . 40 COLOR = 6
5. Delete line 40 40
6. Add a point that connects 5,7 with 5,5 . . . . 60 PLOT ?,?

Don Piele, University of Wisconsin-Parkside, Kenosha, WI 53141.

School, con't...

7. Add line 30 ......30 PLOT 39,39
8. Add line 40 ......40 PLOT 40,40
9. Delete line 40 40
10. Write a program that will display the first letter of your first name in graphics.

Lesson #5

Key Words: FOR-NEXT

LIST	EXPLANATION
10 GR 20 FOR I = o TO 15	Begin a loop with I = 0 and increase I by one each time until I = 15.
30 COLOR = I	The color changes with each pass through the loop.
40 PLOT I,15	The position to be plotted changes with each pass.
50 NEXT I	End of the loop. Go back to statement 20 if I is less than 15. Otherwise go to line 60.
100 END	
RUN	

Your Turn (RUN the program after each change.)

1. Change line 40 to ....40 PLOT I,10
2. Change line 40 to ....40 PLOT 20,I
3. Change line 20 to ....20 FOR I = 0 TO 39
4. Change line 40 to .....40 PLOT I,I
5. Add line 45 ....45 PLOT 39-I,I
6. Change the program to draw + in graphics.

Lesson #10

Key Words: RND, IF-THEN

LIST	EXPLANATION
10 GR 20 COLOR = 9 30 X = RND (40)	A random number is chosen from the numbers 0 to 39 and put in X.
40 Y = RND (40)	Another random number in chosen and placed in Y.
50 IF Y>20 THEN COLOR = 3	If Y is larger than 20 then change the color to blue(3).
60 PLOT X,Y 70 GOTO 20 100 END	Go to line 20 and repeat.
RUN	

Your Turn (RUN the program after each change.)

1. Change 50 to ....50 IF Y > 10 THEN COLOR = 3
2. Change 70 to ....70 GOTO 30
3. Change 70 to ....70 GOTO 20
4. Change 50 to ....50 IF X > 20 THEN COLOR = 3
5. Add 55.........55 IF Y > 20 THEN COLOR = 13
6. Delete 55 55
7. Change 50 ....50 IF X + Y > 40 THEN COLOR = 3
8. Adjust the program to plot 4 different colors in the four different corners of the screen.

Tell & Run

In addition to the lessons, the students were given short problems to solve. They were asked to predict the output of a given program before they observed it run on the TV screen. This gave the students a chance to test their ability to reason sequentially through the statements of a program. A few examples are given below.

#1	#2
10 GR	10 GR
20 FOR I = 0 TO 10	20 FOR I = 10 to 20
30 COLOR = 9	30 COLOR = I
40 PLOT 2,I	40 HLIN 0,39 AT I
50 NEXT I	50 NEXT I
60 END	60 END

The original purpose for doubling up was to provide more computer time for the class each week. But it turned out to be valuable for a completely different reason - cooperation.

Other activities reversed the process and presented a picture and asked the student to write a program that would produce the same result. Here are a few examples:

Computer As A Creative Tool

The APPLE II microcomputer was left in the classroom during the week to give the class time to experiment. Students signed up in pairs to work on the exercises together. The original purpose for doubling up was to provide more computer time for the class each week. But it turned out to be valuable for a completely different reason - cooperation. The students helped each other figure out the effect of each new command. The programming exercises facilitated discussions about the behavior of each new statement. New discoveries were shared with pride and enthusiasm. The computer was the focus and facilitator for cooperative problem-solving.

Student Reactions

After eight weeks, the students were asked to respond to the following questionnaire, using a scale of 1 to 5 (1-strongly disagree, 5-strongly agree), with responses from the 6th grade class of 14 boys and 10 girls recorded.

Reflections

Only a small sample of the exercises done by the students are presented in this article. An entire collection of problems was prepared for the 6th grade class to be used on the APPLE II. At the present time, good materials are not readily available. This presents a formidable obstacle to the inexperienced teacher who wants to use computers in the classroom. As more classrooms begin using microcomputers and sharing their work with others, this problem will diminish.

Students in this sixth grade class were very enthusiastic about working with a microcomputer. In contrast, students at a nearby high school who had not been exposed to computers before were generally uninterested in learning how to use them. Perhaps by this time, the older students have other activities that are more relevant. Also, in the sixth grade the survey shows that boys and girls are equally confident and interested in programming the computer. However, in entries from 10th graders in an annual computer problem-solving contest held at UW-Parkside, the boys outnumber the girls 9 to 1. A recent survey in Creative Computing Magazine had a response with a distribution of 95.4% male and 4.2% female.

Programming the microcomputer was considered by the sixth graders to be highly motivating. They would rather spend their recess on it than go outdoors. Students came early to school and would hang around as long as they could after school. A sign-up sheet became a necessity. With practice, some of the students became resident 'experts' able and thrilled to help others - including the teacher. Some of the sixth graders entered our annual computer programming contest.

School, con't...

Research Questions

This pilot project suggests a number of possible topics for further investigation and research:

1. If students learn how to program a computer early, will they maintain their enthusiasm in later years?
2. If students learn how to program a computer early, will the interest and confidence level of girls, in later years, continue to match that of boys?
3. What factors influence the acceptance of microcomputers in the classroom? Graphics? Games? Programming problems?
4. What is the relationship between logical thinking skills and creative programming skill?

Conclusion

Computers have been used in education primarily as a **delivery system** for subject matter. This role will continue to be developed even further with microcomputers. However, a new application is emerging which is fundamentally different. Instead of the computer programming the student, the student learns how to program the computer. Arthur Luehmann describes it as follows:

Instead of the computer programming the student, the student learns how to program the computer.

> "Computing constitutes a new and fundamental intellectual resource. To use that resource as a mere delivery system for instruction, but not to give a student instruction in how he/she might use the resource, has been the chief failure of the CAI effort. What a loss of opportunity if the skill of computing were to be harnessed for the purpose of turning out masses of students who are unable to use computing."

The computer as an instrument for learning logical thinking and problem-solving skills is only beginning to be understood. However, with the rapid development of low cost microcomputers in the next few years, computers - and hence computer problem-solving techniques - will become a fundamental intellectual resource. □

APPLE II® PROFESSIONAL SOFTWARE

PIE TEXT EDITOR

PIE (PROGRAMMA IMPROVED EDITOR) is a two-dimensional cursor-based editor designed specifically for use with memory-mapped and cursor-based CRT's. It is totally different from the usual line-based editors, which were originally designed for Teletypes. The keys of the system input keyboard are assigned specific PIE Editor function commands. Some of the features included in the PIE system are: Blinking Cursor; Cursor movement up, down, right, left, plus tabs; Character insert and delete; String search forwards and backwards; Page scrolling; GOTO line number, plus top or bottom of file; Line insert and delete anywhere on screen; Move and copy (single and multiple lines); Append and clear to end of line; Efficient memory usage. The following commands are available in the PIE Text Editor and each is executed by depressing the systems argument key simulataneously with the command key desired:

[LEFT]	Move cursor one position to the left
[RGHT]	Move cursor one position to the right
[UP]	Move cursor up one line
[DOWN]	Move cursor down one line
[BHOM]	Home cursor in lower left left hand corner
[HOME]	Home cursor in upper left hand corner
[-PAG]	Move up (toward top of file) one "page"
[+PAG]	Move down (toward bottom of file) one "page"
[LTAB]	Move cursor left one horizontal tab
[RTAB]	Move cursor right one horizontal tab
[GOTO]	Go to top of file (line 1)
[ARG]n[GOTO]	Go to line 'n'
[BOT]	Go to bottom of file (last line + 1)
[-SCH]	Search backwards (up) into file for the next occurence of the string specified in the last search command
[ARG]t[-SCH]	Search backwards for string 't'
[+SCH]	Search forwards (down) into the file for the next occurence of the string specified in the last search command
[ARG]t[+SCH]	Search forward for string 't'
[APP]	Append -move cursor to last character of line +1
[INS]	Insert a blank line beforere the current line
[ARG]n[INS]	Insert 'n' blank lines before the current line
[DEL]	Delete the current line, saving it in the "push" buffer
[ARG]n[DEL]	Delete 'n' lines and save the first 20 in the "push" buffer
[DBLK]	Delete the current line as long as it is blank
[PUSH]	Save current line in "push" buffer
[ARG]n[PUSH]	Save 'n' lines in the "push" buffer
[POP]	Copy the contents of the "push" buffer before the current line
[CINS]	Enable character insert mode
[CINS] [CINS]	Turn off character insert mode
[BS]	Backspace
[GOB]	Gobble - delete the current character and pull remainder of characters to right of cursor left one position
[EXIT]	Scroll all text off the screen and exit the editor
[ARG] [HOME]	Home Line - scroll up to move current line to top of screen
[APP] [APP]	Left justify cursor on current line
[ARG] [GOB]	Clear to end of line

Apple PIE Cassette	**16K**	**$19.95**
TRS-80PIE Cassette	**16K**	**19.95**
Apple PIE Disk	**32K**	**24.95**

6502FORTH · Z-80FORTH 6800 FORTH

FORTH is a unique threaded language that is ideally suited for systems and applications programming on a micro-processor system. The user may have the interactive FORTH Compiler/Interpreter system running standalone in 8K to 12K bytes of RAM. The system also offers a built-in incremental assembler and text editor. Since the FORTH language is vocabulary based, the user may tailor the system to resemble the needs and structure of any specific application. Programming in FORTH consists of defining new words, which draw upon the existing vocabulary, and which in turn may be used to define even more complex applications. Reverse Polish Notation and LIFO stacks are used in the FORTH system to process arithmetic expressions. Programs written in FORTH are compact and very fast.

SYSTEM FEATURES & FACILITIES

Standard Vocabulary with 200 words
Incremental Assembler
Structured Programming Constructs
Text Editor
Block 1/0 Buffers
Cassette Based System
User Defined Stacks
Variable Length Stacks
User Defined Dictionary
Logical Dictionary Limit
Error Detection
Buffered Input

CONFIGURATIONS

AppleFORTH Cassette 16K	**$34.95**
AppleFORTH Disk 32K	**49.95**
PetFORTH Cassette 16K	**34.95**
TRS-80FORTH Cassette 16K	**34.95**
SWTPCFORTH Cassette 16K	**34.95**

ASM/65 EDITOR ASSEMBLER

ASM/65 is a powerful, 2 pass disk-based assembler for the Apple II Computer System. It is a compatible subset of the FORTRAN cross-assemblers which are available for the 6500 family of micro-processors. ASM/65 features many powerful capabilities, which are under direct control of the user. The PIE Text Editor co-resides with the ASM/65 Assembler to form a comprehensive development tool for the assembler language programmer. Following are some of the features available in the ASM/65 Editor Assembler.

PIE Text Editor Command Repetoire
Disk Based System
Decimal, Hexadecimal, Octal, & Binary Constants
ASCII Literal Constants
One to Six character long symbols
Location counter addressing "*"
Addition & Subtraction Operators in Expressions
High-Byte Selection Operator
Low-Byte Selection Operator
Source statements of the form:
[label] [opcode] [operand] [;comment]
56 valid machine instruction mnemonics
All valid addressing modes
Equate Directive
BYTE Directive to initialize memory locations
WORD Directive to initialize 16-bit words
PAGE Directive to control source listing
SKIP Directive to control source listing
OPT Directive to set select options
LINK Directive to chain multiple text files
Comments
Source listing with object code and source statements
Sorted symbol table listing

CONFIGURATION

Apple II	**48K/Disk**	**$69.95**

LISA INTERACTIVE ASSEMBLER

LISA is a totally new concept in assembly language programming. Whereas all other assemblers use a separate or co-resident text editor to enter the assembly language program and then an assembler to assemble the source code, LISA is fully interactive and performs syntax/addressing mode checks as the source code is entered in. This is similar in operation to the Apple II Integer BASIC Interpreter. All error messages that are displayed are in plain, easy to understand English, and not simply an Error Code. Commands in LISA are structured as close as possible to those in BASIC. Commands that are included are: LIST, DELETE, INSERT, PR #n, IN #n, SAVE, LOAD, APPEND, ASM, and a special user-defineable key envisioned for use with "dumb" peripherals. LISA is DISK II based and will assemble programs with a textfile too long to fit into the Apple memory. Likewise, the code generated can also be stored on the Disk, hence freeing up memory for even larger source programs. Despite these Disk features, LISA is very fast; in fact LISA is faster than most other commercially available assemblers for the Apple II. Not only is LISA faster, but also, due to code compression techniques used LISA requires less memory space for the text file. A full source listing containing the object and source code are produced by LISA, in addition to the symbol table

Apple II 32K/Disk $34.95

PROGRAMMA

Software Products

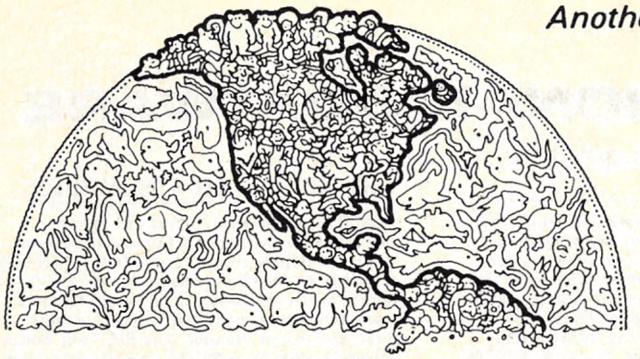
PROGRAMMA INTERNATIONAL, INC.

3400 Wilshire Blvd.
Los Angeles, CA 90010

(213) 384-0579 • 384-1116 • 384-1117

Apple II is a registered trademark of Apple Computers, Inc. These professional products are available at your local computer dealer.

Another new game from Creative Computing . . . .



Geography

Jonathan Amsterdam

Geography is a simple two-player game which has been adapted to a game against the computer. The instructions in the sample run explain how to play the game.

The only problem in writing a computer program for Geography was the list of geographical names. In the actual person vs. person game, any geographical location can be used. I chose to restrict possible choices to the categories listed in the instructions. This results in about three hundred pieces of data—sufficient for a challenging game but not enough to appreciably slow down the computer. The list can, of course, be extended by including categories such as world's highest mountains, world's longest rivers, state counties, etc. If the game begins to slow down, you could arrange the data in alphabetical order and use a more efficient search, for example, a binary search. I use a sequential search, with the advantage of being able to add new data right onto the end of the program.

Geography was written on a PDP-11/70 with RSTS/E BASIC. String functions may present the only problem in translation. In RSTS/E BASIC, for some reason, LEFT(A$,N) will take the N leftmost characters of A$, but RIGHT(A$,N) will take the right part of A$ starting from the Nth character.

Important variables: A is the number of data elements; when adding data, it is necessary only to change the value of A (line 30) and the DIM statement (line 40). Everything else will take care of itself. B$ holds the data; U$ holds the items already used in the game; U is the counter for U$ and C indicates whose turn it is.

Most of the code is self-explanatory; the only confusing part might be the search in lines 400-500. To prevent the program from choosing the same response in every game, the search for a match begins at a random point in the middle of the array, proceeds to the end, then goes from the beginning to the original random point. If no match is found, the player wins; otherwise, the response is printed and it is the player's turn.

Finally, for those lucky few with voice synthesizers, Geography would be an ideal application of computer speech. □

Jonathan Amsterdam, 848 Hayes St., Baldwin, NY 11510.

```
INSTRUCTIONS? YES
THIS IS A GAME OF 'GEOGRAPHY.   ONE PLAYER NAMES A
GEOGRAPHICAL LOCATION.  THE OTHER PLAYER MUST NAME
A LOCATION WHOSE FIRST LETTER IS THE SAME AS THE
LAST LETTER OF THE PRECEEDING WORD.  YOU WILL PLAY
AGAINST THE COMPUTER.  ONLY THE FOLLOWING
CATEGORIES WILL BE USED:

                         CONTINENTS
                         COUNTRIES
                         U.S. STATES
                         CANADIAN PROVINCES
                         MAJOR WORLD CITIES
                         U.S. STATE CAPITALS
                         OCEANS AND SEAS

THE COMPUTER WILL TELL YOU IF THE LETTERS OF
YOUR ENTRY AND THE PRECEEDING WORD DON'T MATCH,
THE WORD HAS ALREADY BEEN USED, OR THE WORD IS
NOT IN THE COMPUTER'S MEMORY ('ILLEGAL WORD').
IF YOU CAN'T THINK OF A CORRECT RESPONSE,
YOU MAY TYPE 'CHALLENGE.'  IF YOU'VE REALLY
EXHAUSTED ALL POSSIBLE RESPONSES, YOU'VE WON. OTHERWISE,
THE COMPUTER WILL PRINT OUT A POSSIBLE RESPONSE, AND
YOU LOSE.
DO YOU WANT TO GO FIRST? NO
ZAIRE
? EAST GERMANY
YEMEN
? NEW YORK
KENYA
? ALABAMA
ATHENS
? SALT LAKE CITY
YELLOW SEA
? ASIA
AMSTERDAM
? MISSISIPPI
*** ILLEGAL WORD ***
? MISSISSIPPI
INDIANA
? AUGUSTA
ANNAPOLIS
? SOUTH CAROLINA
ATLANTA
? AUSTRIA
ALBANY
? YUGOSLAVIA
AUSTIN
? NORTH AMERICA
ATLANTIC OCEAN
? NORTH DAKOTA
ARCTIC OCEAN
? NORTH CAROLINA
ARKANSAS
? SOUTH DAKOTA
ARIZONA
? ALBANY
*** ALREADY USED ***
? CHALLENGE
ALASKA
YOU LOSE
PLAY AGAIN? NO
```

```
1 REM  **********************
2 REM  *      GEOGRAPHY     *
3 REM  *                    *
4 REM  * JONATHAN AMSTERDAM *
5 REM  *  848 HAYES STREET  *
6 REM  *  BALDWIN NY 11510  *
7 REM  *                    *
8 REM  *     JUNE 1979      *
9 REM  **********************
20 RANDOMIZE
30 A=345
40 DIM B$(345),U$(345)
50 FOR I=1 TO A\READ B$(I)\NEXT I
60 PRINT "INSTRUCTIONS";\INPUT A$\IF A$<>"YES" THEN 230
70 PRINT "THIS IS A GAME OF 'GEOGRAPHY.'  ONE PLAYER NAMES A"
80 PRINT "GEOGRAPHICAL LOCATION.  THE OTHER PLAYER MUST NAME"
90 PRINT "A LOCATION WHOSE FIRST LETTER IS THE SAME AS THE"
100 PRINT "LAST LETTER OF THE PRECEEDING WORD.  YOU WILL PLAY"
110 PRINT "AGAINST THE COMPUTER.  ONLY THE FOLLOWING "
```

COMPUTER RAGE GAME

50% discount on the purchase of 5 (or more) Games

Discount Price

5 for $22.38

Additional games $4.49 each

Regular price

5 for $44.75
$8.95 each

Computer Rage has been hailed by educators as an outstanding game for teaching youngsters between 7 and 14 about the binary number system (the game uses 3 binary dice!), parts of a computer system and how a program is processed. In addition the game is sheer fun! Recommended by Instructor, The Arithmetic Teacher, The Science Teacher, Curriculum Product Review and others.

Computer Rage is based on a large'scale multiprocessing computer system. The objective is to move your three programs from input to output. Moves are determined by the roll of three binary dice representing bits in a computer. Hazards include priority interrupts, program bugs, decision symbols, power failures and restricted input and output channels. Notes are included for adapting game for school instruction. A perfect introductory tool to binary math and the seemingly-complex computer. [6Z]

Binary Dice. Now, the same dice used in Computer Rage can be purchased separately. Three binary dice (red, green and blue) in a zip-lock bag. $1.25 postpaid. [3G]

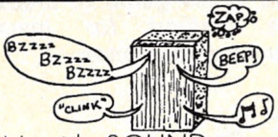
To Order Send order with payment to **Creative Computing** P.O. Box 789-M, Morristown, NJ 07960. Visa, Master Charge or American Express are acceptable.
For faster service, call in your bank card order toll free to: **800-631-8112** (in NJ call 201-540-0445)

```
120 PRINT "CATEGORIES WILL BE USED:"
130 PRINT\FOR I=1 TO 7\READ C$\PRINT TAB(25);C$\NEXT I\PRINT
140 PRINT "THE COMPUTER WILL TELL YOU IF THE LETTERS OF"
150 PRINT "YOUR ENTRY AND THE PRECEEDING WORD DON'T MATCH,"
160 PRINT "THE WORD HAS ALREADY BEEN USED, OR THE WORD IS"
170 PRINT "NOT IN THE COMPUTER'S MEMORY ('ILLEGAL WORD')."
180 PRINT "IF YOU CAN'T THINK OF A CORRECT RESPONSE,"
190 PRINT "YOU MAY TYPE 'CHALLENGE.'  IF YOU'VE REALLY"
200 PRINT "EXHAUSTED ALL POSSIBLE RESPONSES, YOU'VE WON. OTHERWISE,"
210 PRINT "THE COMPUTER WILL PRINT OUT A POSSIBLE RESPONSE, AND"
220 PRINT "YOU LOSE."
230 U,C=0\PRINT "DO YOU WANT TO GO FIRST";\INPUT A$
240 PRINT\IF A$="NO" THEN 250
245 C=1\GOTO 260
250 R$=CHR$(INT(RND*26+65))\GOTO 400
255 C=0
260 INPUT A$\IF C=1 THEN 300
270 IF A$="CHALLENGE" THEN 550
280 IF LEFT(A$,1)=R$ THEN 300
290 PRINT "*** LETTERS DON'T MATCH ***"\GOTO 260
300 FOR E=1 TO U
310     IF A$<>U$(E) THEN 330
320     PRINT "*** ALREADY USED ***"\GOTO 260
330 NEXT E
340 FOR E=1 TO A
350     IF A$<>B$(E) THEN 370
360     GOTO 390
370 NEXT E
380 PRINT "*** ILLEGAL WORD ***"\GOTO 260
390 R$=RIGHT(A$,LEN(A$))\U=U+1\U$(U)=A$
400 G=INT(RND*A+1)\H=A
410 FOR E=G TO H
420     IF LEFT(B$(E),1)<>R$ THEN 480
430     FOR I=1 TO U
440             IF B$(E)=U$(I) THEN 480
450     NEXT I
460     PRINT B$(E)\U=U+1\U$(U)=B$(E)
470     R$=RIGHT(B$(E),LEN(B$(E)))\GOTO 255
480 NEXT E
490 IF G=1 THEN 510
500 H=G\G=1\GOTO 410
510 PRINT "YOU WIN"
520 PRINT "PLAY AGAIN";
530 INPUT A$\IF A$="YES" THEN 230
540 GOTO 1350
550 REM *** CHALLENGE SUBROUTINE ***
560 FOR E=1 TO A
570     IF LEFT(B$(E),1)<>R$ THEN 620
580     FOR I=1 TO U
590             IF B$(E)=U$(I) THEN 620
600     NEXT I
610     PRINT B$(E)\PRINT "YOU LOSE"\GOTO 520
620 NEXT E\GOTO 510
630 DATA "NEW YORK","NEW JERSEY","PENNSYLVANIA","CONNECTICUT","ALABAMA"
640 DATA "VERMONT","NEW HAMPSHIRE","MAINE","FLORIDA","MASSACHUSETTS"
650 DATA "GEORGIA","SOUTH CAROLINA","RHODE ISLAND","UTAH","INDIANA"
660 DATA "NORTH CAROLINA","VIRGINIA","SOUTH DAKOTA","OHIO","OREGON"
670 DATA "NEW MEXICO","NORTH DAKOTA","WISCONSIN","MICHIGAN","IDAHO"
680 DATA "CALIFORNIA","MISSISSIPPI","KANSAS","NEBRASKA","IOWA"
690 DATA "WASHINGTON","MONTANA","MINNISOTA","KENTUCKY","TENNESSEE"
700 DATA "LOUISIANA","DELAWARE","MARYLAND","ARKANSAS","COLORADO"
710 DATA "ARIZONA","WEST VIRGINIA","TEXAS","NEVADA","OKLAHOMA"
720 DATA "WYOMING","MISSOURI","ALASKA","HAWAII","ILLINOIS"
730 DATA "ALBERTA","MANITOBA","SASKATCHEWAN","QUEBEC","ONTARIO"
740 DATA "PRINCE EDWARD ISLAND","NEW BRUNSWICK","NEWFOUNDLAND"
750 DATA "AFGHANISTAN","AFRICA","ALBANIA","ALGERIA","ANDORRA"
```

```
760 DATA "ANGOLA","ANTARCTICA","ARGENTINA","ASIA","AUSTRALIA"
770 DATA "AUSTRIA","BAHRAIN","BANGLADESH","BARBADOS","BELGIUM"
780 DATA "BHUTAN","BOLIVIA","BOTSWANA","BRAZIL","BRITISH HONDURAS"
790 DATA "BULGARIA","BURMA","BURUNDI","CAMBODIA","CAMEROON"
800 DATA "CANADA","CENTRAL AFRICAN REPUBLIC","CEYLON","CHAD","CHILE"
810 DATA "CHINA","COLUMBIA","CONGO","COSTA RICA","CUBA"
820 DATA "CYPRUS","CZECHOSLOVAKIA","DAHOMEY","DENMARK","DOMINICAN REPUBL
IC"
830 DATA "ECUADOR","EGYPT","EL SALVADOR","ENGLAND","ETHIOPIA"
840 DATA "EQUATORIAL GUINEA","EUROPE","FIJI","FINLAND","FRANCE"
850 DATA "GABON","GAMBIA","EAST GERMANY","WEST GERMANY","GERMANY"
860 DATA "GHANA","GREAT BRITAIN","GREECE","GUATEMALA","GUINEA"
870 DATA GUYANA,HAITI,HOLLAND,HONDURAS,HUNGARY
880 DATA "ICELAND","INDIA","INDONESIA","IVORY COAST","IRAN"
890 DATA "IRAQ","IRELAND","ISRAEL","ITALY","JAMAICA"
900 DATA "JAPAN","JORDAN","KENYA","SOUTH KOREA","NORTH KOREA"
910 DATA "KUWAIT","LAOS","LEBANON","LESOTHO","LIBERIA"
920 DATA "LIBYA","LIECHTENSTEIN","LUXEMBOURG","MALAWI","MALAYA"
930 DATA "MALAGASY REPUBLIC","MALAYSIA","MALDIVES","MALI","MALTA"
940 DATA "MAURITANIA","MAURITIUS","MEXICO","MONGOLIA","MOZAMBIQUE"
950 DATA "NEPAL","NETHERLANDS","NEW ZEALAND","NICARAGUA","NIGER"
960 DATA "NIGERIA","NORWAY","OMAN","PAKISTAN","PANAMA"
970 DATA "PARAGUAY","PERU","PHILIPPINES","POLAND","PORTUGAL"
980 DATA "QATAR","RHODESIA","RUMANIA","RWANDA","SAUDI ARABIA"
990 DATA "SCOTLAND","SENEGAL","SIERRA LEONE","SINGAPORE","SOMALIA"
1000 DATA "SOUTH AFRICA","SOUTH AMERICA","NORTH AMERICA","SPAIN","SUDAN"
1010 DATA "SURINAM","SWAZILAND","SWEDEN","SWITZERLAND","SYRIA"
1020 DATA "TANZANIA","THAILAND","TOGO","TRINIDAD","TUNISIA"
1030 DATA "UNION OF SOVIET SOCIALIST REPUBLICS","WALES","ZAIRE","YEMEN",
"ZAMBIA"
1040 DATA "UPPER VOLTA","URUGUAY","VATICAN CITY","VENEZUELA","UNITED KIN
GDOM"
1050 DATA "NORTH VIETNAM","SOUTH VIETNAM","RUSSIA","UNITED STATES","YUGO
SLAVIA"
1060 DATA "TOKYO","LONDON","SHANGHAI","OSAKA","MOSCOW"
1070 DATA "BUENOS AIRES","PARIS","MEXICO CITY","CHICAGO","LOS ANGELES"
1080 DATA "CALCUTTA","PEKING","SAO PAULO","BOMBAY","DJAKARTA"
1090 DATA "SEOUL","RIO DE JANEIRO","CAIRO","PHILADELPHIA","DETROIT"
1100 DATA "HONG KONG","BERLIN","LENINGRAD","MANILA","NEW DEHLI"
1110 DATA "BOSTON","KARACHI","MADRID","SAN FRANCISCO","WASHINGTON DC"
1120 DATA "SYDNEY","ROME","MILAN","SANTIAGO","MONTREAL"
1130 DATA "LIMA","BARCELONA","MELBOUNE","PITTSBURG","MANCHESTER"
1140 DATA "ST. LOUIS","TORONTO","HAMBURG","BUDAPEST","CLEVELAND"
1150 DATA "BANGKOK","JOHANNESBURG","ATHENS","SINGAPORE","ISTANBUL"
1160 DATA "VIENNA","BRUSSELS","AMSTERDAM","GLASGOW","HAVANA"
1170 DATA "LISBON","WARSAW","COPENHAGEN","STOCKHOLM","PRAGUE"
1180 DATA "MONTGOMERY","JUNEAU","PHOENIX","LITTLE ROCK","SACRAMENTO"
1190 DATA "DENVER","HARTFORD","DOVER","TALLAHASSEE","ATLANTA"
1200 DATA "HONOLULU","BOISE","SPRINGFIELD","INDIANAPOLIS","DES MOINES"
1210 DATA "TOPEKA","FRANKFORT","BATON ROUGE","AUGUSTA","ANNAPOLIS"
1220 DATA "LANSING","ST. PAUL","JACKSON","JEFFERSON CITY","HELENA"
1230 DATA "LINCOLN","CARSON CITY","CONCORD","TRENTON","SANTA FE"
1240 DATA "ALBANY","RALEIGH","BISMARCK","COLUMBUS","OKLAHOMA CITY"
1250 DATA "SALEM","HARRISBURG","PROVIDENCE","COLUBMIA","PIERRE"
1260 DATA "NASHVILLE","AUSTIN","SALT LAKE CITY","MONTPELIER","RICHMOND"
1270 DATA "OLYMPIA","CHARLESTON","MADISON","CHEYENNE","PACIFIC OCEAN"
1280 DATA "ATLANTIC OCEAN","INDIAN OCEAN","ARCTIC OCEAN","RED SEA","BLAC
K SEA"
1290 DATA "MEDITERRANEAN SEA","BERING SEA","NORTH SEA","BALTIC SEA","YEL
LOW SEA"
1300 DATA "CARIBBEAN SEA","GULF OF MEXICO","EAST CHINA SEA","HUDSON BAY"
1310 DATA "SEA OF JAPAN","SOUTH CHINA SEA","XENIA","XANTHUS"
1320 DATA "YALTA","YANGTZE RIVER","YOKOHAMA","YUKON TERRITORY"
1330 DATA "CONTINENTS","COUNTRIES","U.S. STATES","CANADIAN PROVINCES"
1340 DATA "MAJOR WORLD CITIES","U.S. STATE CAPITALS","OCEANS AND SEAS"
1350 END
```

realsoft™ THE PROGRAM STORE

(202) 337-4691
4200 Wisconsin Ave. N.W. Washington D.C. 20016



ALL with SOUND ************

Sound effects and music are new for the TRS-80. You will find below programs with sound which will entertain both you and your friends. All require Level II and 16k, except Star Warp which only needs 4k.

bee wary
by Leo Christopherson from 80-U.S.
Said the spider to the bee . . . Try to survive as the bee against the spider. By author of Android Nim. $14.95

music
by Tom Stibolt from Acorn
Enter scores to have your TRS-80 play music. Works with both disk and tape. The "Sting" theme provided on the tape. $9.95

morse
by Phil Pilgrim from Discovery Bay
Ever want to learn Morse code? Here's your chance. Interactive program provides more rapid mastering of material. $14.95

yatch race
by John Greenwood from Mad Hatter
Turn-of-the-century board game of skill and strategy. Sail around three buoys avoiding islands and peninsulas. $7.95

life two
by Leo Christopherson from 80-U.S.
Two programs. Battle of Life allows several players and has noisy creatures. Other version is Conway's Life at 100 generations per minute. $14.95

star warp+
by Mike Partain and Ray Daly from Acorn
Star Warp requires real-time positioning of cross-hairs and includes a timer and scoring. Tape includes Graphic Lunar Lander program. Both have sound. $9.95

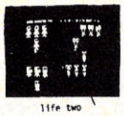
mind boggle
by John Greenwood from Mad Hatter
New, real-time word game for all ages. Guess hidden word based on "cows" and "bulls" (clues) and sound. $7.95

star trek
by John Christman from Acorn
Non-scrolling display and sound make this version interesting for any Captain. Real time and moving Klingons. $9.95

snake eggs
by Leo Christopherson from 80-U.S.
Talking snakes and you play a version of '21'. Animated, songs . . . almost a cartoon. $14.95

bandito
by Doug Prousser from Acorn
Take a gamble. Pull the arm on this graphic slot machine with sound. $9.95

android nim
by Leo Christopherson from 80-U.S.
Animated, squeaking androids play well known game of Nim against you. "It is a classic." "The best demo tape for Level II . . ." - Purser. $14.95


life two


snake eggs

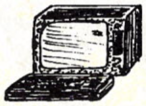

android nim

****

gammon challenger™
by Ray Daly and Tom Throop
This backagammon playing program has three levels of play and a variety of special features. They include saving and recalling board positions, setting up a particular board position, switching sides with the program, letting the program play again, and changing the level of play.
Challenger has played quite well against other programs and dedicated games. In fact, it was the subject of a recent story in Personal Computing. Enjoy playing backgammon today. 16k Level II $14.95

IF YOUR COMPUTER LIKES PROGRAMS, YOU WILL LIKE US. WE CARRY MORE PROGRAMS FOR THE RADIO SHACK TRS-80 COMPUTER THAN ANYONE. WE ARE DEALERS FOR OVER THIRTY DIFFERENT COMPANIES AND WE CONTINUOUSLY SEARCH FOR EXCITING, NEW PROGRAMS.

YOU MAY HAVE NOTICED OUR CHANGE OF NAME. FORMERLY COMPUTER CABLEVISION WE HAVE JUST MOVED INTO OUR NEW STORE IN WASHINGTON, D.C. WE HAVE NOT ONLY TRS-80 SOFTWARE, BUT A VARIETY OF EQUIPMENT. PLEASE VISIT.



TRS-80
a trademark of Tandy Corporation

electric pencil
from Michael Shrayer
Considered the best word processing program available for TRS-80. Extremely easy to use, Pencil lets you produce mailing lists, forms, large numbers of original correspondence, and camera ready copy for printing.
The editor of Creative Computing said that it "will open a whole new dimension for any writer." 16k Level I or II $99. Disk version $150.

editor
by Dick Bowles from Software Associates
Word processing with upper and lower case without any hardware modifications to the TRS-80. Insert or change lines, search for words or phrases, print whole or partial text, save and retrieve on disk, center lines, right justification. Requires 32k and disk. $39.95

****

SIMULATIONS

santa paravia
by Rev. George Blank from Instant
As ruler you struggle to create a kingdom based on how well you manage your medieval city-state. Up to six players compete to become King or Queen. 16k Level I or II $7.95

oil tycoon
by Dwight Meyer from Instant
Two player compete in the search for the "black gold." Invest in research and take your chances on geologists' reports. Once you strike oil, you try to control the market. 4k Level I or 16k Level II $7.95

population
from Instant Software
Two players either cooperate or compete in trying to make a better world on Techno and Primo. Also includes Stellar Wars game. 16k Level II $7.95

pork barrel
by Rev. George Blank from Mad Hatter
Play a Congressman deciding how to vote on sensitive issues without losing your constituency. 16k Level II $7.95

stock market
by Andrew Bartorillo from Acorn
Upto four players invest in stocks with buying on margin allowed under current SEC regulations. Random volatility, news flashes, stock splits, and bull and bear markets. 16k Level II $9.95

SYSTEM SAVERS
by Tom Stibolt from Acorn
If you ever use the SYSTEM command, you should buy this two program package. These programs allow you to save any system tape on tape or disk, plus offer several features for machine language programmers.
In Computer Cassettes Magazine, Robert Purser said that a tape duplication program "should be in everyone's library." We agree and beleive that System Savers is the perfect answer.
With FLEXL, which is one of the two programs, you can make back-up copies of any system format tape. Most often a cassette that you make will load easier than an orginal. This also protects the valuable copies of your system programs.
Plus you can find the filename on any system tape because it is displayed on the screen. If you ever forget to save a program filename, just use FLEXL to recover it. And at anytime you can stop the reading of the tape by simply pressing BREAK.
For any machine language programmer, FLEXL offers the advantage of producing more efficient tapes than the object files from the assembler. It is also written to interface directly with the Small Systems RSM-1 and other monitor programs. And machine language tapes can be merged following the prompting of the program.
Disk drive owners can use TDISK to save any system format tape onto disk. Sargon, Adventure, Airaid, Editor/Assembler and other programs cannot normally be loaded to disk. Now, TDISK allows you to save these programs onto disk. After DOS READY you will be able to simply type the filename and be up and running. It even loads non-contiguous tapes. TDISK will greatly increase the benefit of owning a disk drive.
And as a FREE BONUS, Acorn Software Products, Inc. also provides complete instructions on how to load MicroChess 1.5 onto disk. A short BASIC program will let you play chess directly from disk.
Complete your system with the routines not found in either Level II or DOS for only $14.95. Order your System Savers, today!

DISASSEMBLER
by Roy Soltoff from Misosys & Acorn
This two pass Z-80 disassembler produces symbolic lables with output to either the video monitor, printer or tape. Radio Shack's Editor/Assembler can load the tapes. If you own the Editor/Assembler, complete the package with this program. Program on tape for two different memory locations. TRS Level II $19.95

QEDIT/AUTOK
by Phil Pilgrim from Discovery Bay
AUTOK adds auto-repeat to any key on your TRS-80. QEDIT is a quick, two-dimensional editor which complements the EDIT function. Both machine language modules include a built-in relocation routine to optimize memory. TRS Level II $14.95
$19.95 version includes single-stoke commands. For example, press SHIFT 'Q' to run a BASIC program.

Credit card callers may call our number listed above. Or clip the coupon and mail your order today.

The Program Store, Dept C
4200 Wisconsin Ave NW
Washington DC 20007

YES, please send these TRS-80 programs

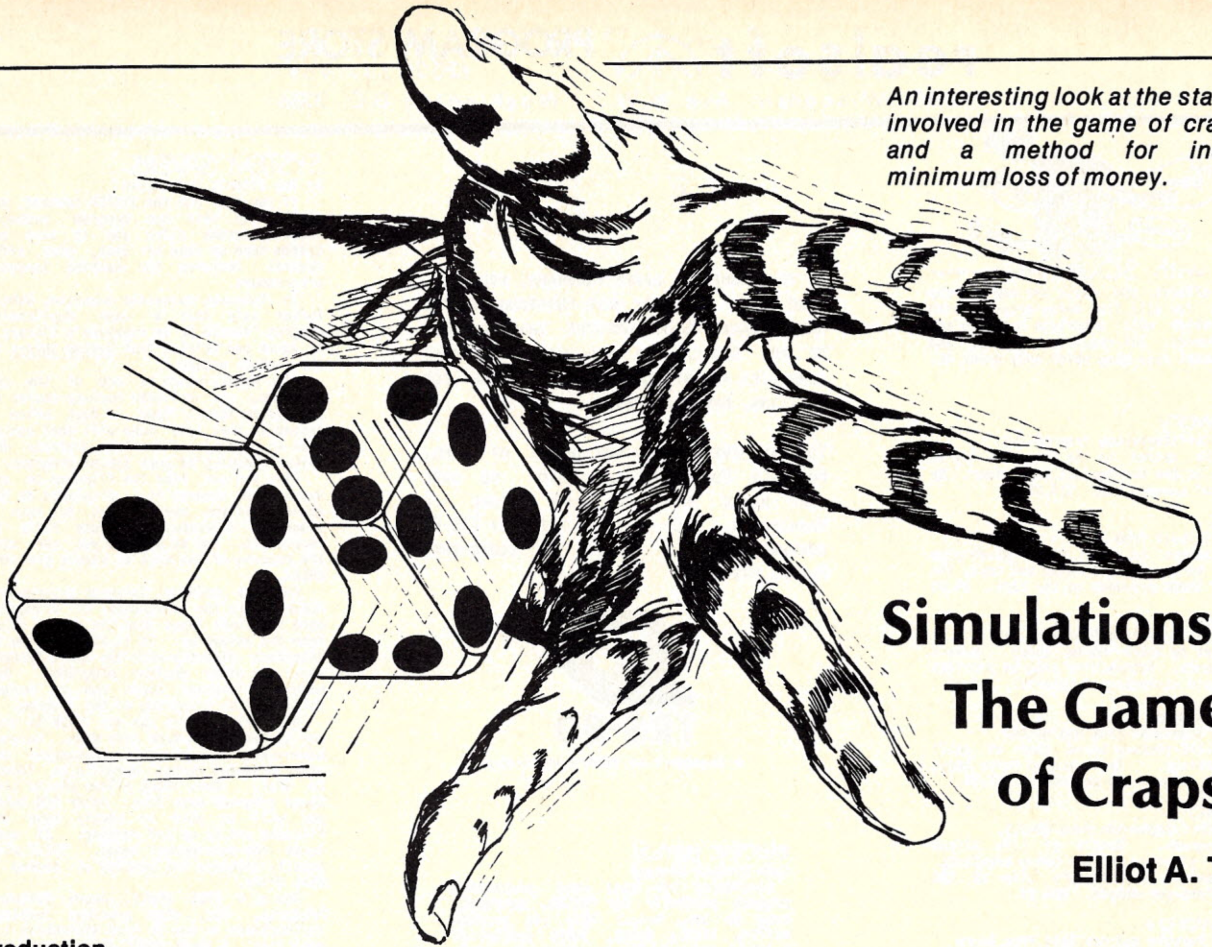
title	price

postage: $ 1.00
total:

name:
address:
city, state & code

☐ Check to the Program Store
☐ MASTERCHARGE mc bank code:
☐ VISA exp date:
card number:
signature:

CIRCLE 182 ON READER SERVICE CARD



An interesting look at the statistics involved in the game of craps ... and a method for insuring minimum loss of money.

Simulations In The Game of Craps

Elliot A. Tanis

Introduction

Almost every state in the United States permits some form of gambling - bingo, horse racing, jai alai, casinos, or lotteries. In some games of chance, the expected loss per play for a gambler is large, in other games the expected loss per play is small.

In this paper we shall use simulation techniques to examine the probability of losing and the expected number of plays before losing $5 in a casino game that is almost fair.

Craps

In the game of craps a pair of dice is rolled and the sum of the spots is observed. The player wins with a 7 or 11, loses with a 2, 3, or 12 on the first roll. If any other number comes up - that's the players "point" and the player continues to roll the dice. If the player's "point" comes up again before a 7 is thrown, the player wins, otherwise the player loses.

Probability of Winning in Craps

The probability that the player wins when rolling the dice in craps is 0.49293. Thus the probability of losing is 0.50707. Since the probability of winning is almost equal to 0.5, this game is almost fair.

Elliot A. Tanis, Math Dept., Hope College, Holland, MI 49423.

The game of craps can be simulated on the computer to illustrate empirically that the probability of winning is 0.49293. A program that can be used for this simulation follows along with a typical run (Listing 1). This program was run on a Tektronix 4051 graphics system. However it can be run on any computer that uses BASIC.

If you run this program on your computer, the proportion of wins should be close to 0.493. However the probability that the proportion of wins is exactly equal to 0.493 is not very large. Probability theory does permit us to say that 95% of the time the proportion of wins in 1000 plays will lie in the interval

$$0.493 - 1.96\sqrt{(0.493)(0.507)/1000}$$

$$\text{to } 0.493 + 1.96\sqrt{(0.493)(0.507)/1000}$$

or 0.462 to 0.524.

If the number of plays is increased to 2000, 95% of the time the proportion of wins will lie in the interval 0.471 to 0.515.

Expected Number of Plays and Rolls

Now suppose that a player has $5 and plans to place $1 bets in the game of craps until this $5 is lost. There are two questions for which we might like to obtain answers using simulation techniques. (1) How many times, on the average, can a player expect to place $1 bet before losing $5? (2) How many times, on the average, can a player expect to roll the dice before losing the $5?

Theoretically it is possible to find the answers. To answer Question 1, we note that for each $1 bet, a dollar is won with probability 0.49293 and a dollar is lost with probability 0.50707. Thus for each bet, a player can expect to lose, on the average,

$$\$1(0.49293) - \$1(0.50707) = \$-0.01414.$$

That is, on the average 1.414 cents is lost on each bet. This means that a player can expect to place on the average,

$$n = \frac{5}{0.01414} = 353.61$$

one dollar bets before losing $5.

To answer Question 2, it can be shown that on the average it requires 3.38 rolls of the dice to determine whether the bettor has won or lost [1]. Thus on the average, it requires (3.38) (353.61) = 1,9.195.20 rolls of the dice to lose $5.

Both of the above answers can be illustrated empirically using simulation techniques on the computer. In addition to illustrating the answers to these two questions in our simulation

Simulations, con't....

program, we shall also find the maximum number of dollars that the player's capital had attained when beginning with $5.

This program (Listing 2) was again written for the Tektronix 4051 graphics system. In line 110, Z9 is for the output device, that is, the screen or plotter.

Summary

Two simulations (Figure 1 and 2) are given which illustrate an interesting phenomenon about the game of craps. Although on the average a player can expect to place 353.61 one dollar bets before losing $5, the simulations show that most of the time, the number of bets placed is less than 353. It requires a "lucky streak" to get a number of plays greater than 353.

Instead of losing $5, it would be nice to know when we have reached our maximum capital. We could then quit and be a winner. Since that information is unknown, we can only guarantee that we are never losers by never gambling. □

Reference

1. Armand V. Smith, Jr., "Some probability problems in the game of 'craps' ", The American Statistician, Vol. 22, No. 3 (1968), pp. 29, 30.

Listing 1

```
100 INIT
110 REM N WILL KEEP TRACK OF THE NUMBER OF WINS.
120 N=0
130 REM SIMULATE 1000 PLAYS IN THE GAME OF CRAPS.
140 FOR K=1 TO 1000
150 REM SIMULATE ROLLING A PAIR OF DICE.
160 REM EACH OF D1 AND D2 CAN EQUAL 1,2,3,4,5, OR 6.
170 REM THEIR OUTCOMES ARE INDEPENDENT.
180 D1=INT(6*RND(-1)+1)
190 D2=INT(6*RND(-1)+1)
200 S1=D1+D2
210 REM THE PLAYER WINS IF S1 = 7 OR 11.
220 IF S1=7 OR S1=11 THEN 350
230 REM THE PLAYER LOSES IF S1 = 2 OR 3 OR 11
240 IF S1=2 OR S1=3 OR S1=12 THEN 360
250 REM IF S1 = 4,5,6,8,9, OR 10, ROLL DICE AGAIN.
260 D1=INT(6*RND(-1)+1)
270 D2=INT(6*RND(-1)+1)
280 S2=D1+D2
290 REM IF S2 = 7, THE PLAYER LOSES
300 IF S2=7 THEN 360
310 REM IF S2 = S1, THE PLAYER WINS
320 IF S2=S1 THEN 350
330 REM OTHERWISE ROLL THE DICE AGAIN
340 GO TO 260
350 N=N+1
360 NEXT K
370 PRINT "THE NUMBER OF WINS OUT OF 1000 PLAYS IS ";N
380 P=N/1000
390 PRINT "THE PROPORTION OF WINS IS ";P
400 END

RUN

THE NUMBER OF WINS OUT OF 1000 PLAYS IS 488
THE PROPORTION OF WINS IS 0.488
```

TRIAL NUMBER	NUMBER OF PLAYS	NUMBER OF ROLLS	MAXIMUM CAPITAL
1	69	224	14
2	17	52	6
3	15	63	7
4	13	42	6
5	9	33	5
6	19	72	6
7	31	120	7
8	95	336	18
9	129	469	13
10	4393	14623	65
11	431	1472	24
12	73	236	12
13	15	54	6
14	11	50	5
15	29	109	6
16	5	18	5
17	129	440	16
18	17	49	6
19	107	329	12
20	1461	4986	39
21	639	2077	35
22	9	47	5
23	7	11	5
24	49	131	9
25	109	402	11
AVERAGES	315.24	1057.80	13.72

Figure 1

Listing 2

```
100 INIT
110 INPUT Z9
120 PAGE
130 REM THIS PROGRAM WILL SIMULATE THE NUMBER OF PLAYS TO
140 REM LOSE $5.  THE SIMULATION IS REPEATED 25 TIMES.
150 REM N WILL HOLD THE NUMBER OF PLAYS.
160 REM R WILL HOLD THE NUMBER OF ROLLS.
170 REM M WILL HOLD THE MAXIMUM CAPITAL.
180 DIM N(25),R(25),M(25)
190 PRINT @Z9:" "
200 PRINT @Z9:" TRIAL     NUMBER     NUMBER    MAXIMUM"
210 PRINT @Z9:" NUMBER   OF PLAYS   OF ROLLS   CAPITAL"
215 PRINT @Z9:
220 FOR K=1 TO 25
230 N(K)=0
240 R(K)=0
250 M(K)=5
260 REM MAXIMUM CAPITAL AT BEGINNING IS $5.
270 REM D WILL EQUAL NUMBER OF DOLLARS HELD BY PLAYER WHO
280 REM BEGINS WITH $5.
290 D=5
300 N(K)=N(K)+1
310 R(K)=R(K)+1
320 REM SIMULATE THE ROLL OF A PAIR OF DICE.
330 D1=INT(6*RND(-1)+1)
340 D2=INT(6*RND(-1)+1)
350 S1=D1+D2
360 IF S1=7 OR S1=11 THEN 490
370 IF S1=2 OR S1=3 OR S1=12 THEN 560
380 REM ROLL DICE AGAIN SINCE S1 = 4,5,6,8,9, OR 10
390 R(K)=R(K)+1
400 D1=INT(6*RND(-1)+1)
410 D2=INT(6*RND(-1)+1)
420 S2=D1+D2
430 REM IF S2 = 7, THE PLAYER LOSES.
440 IF S2=7 THEN 560
450 REM IF S2 = S1, THE PLAYER WINS.
460 IF S2=S1 THEN 490
470 REM OTHERWISE ROLL THE DICE AGAIN.
480 GO TO 390
490 REM PLAYER WINS
500 D=D+1
510 IF D>M(K) THEN 530
520 GO TO 300
530 REM THERE IS A NEW MAXIMUM.
540 M(K)=D
550 GO TO 300
560 REM PLAYER LOSES
570 D=D-1
580 IF D=0 THEN 600
590 GO TO 300
600 REM PLAYER HAS LOST $5.
620 PRINT @Z9: USING 630:" ";K;" ";N(K);"  ";R(K);"  ";M(K)
630 IMAGE 1A,4D,6A,5D,6A,5D,5A,5D
640 NEXT K
650 REM FIND THE AVERAGE OF THE NUMBER OF PLAYS.
660 REM FIND THE AVERAGE OF THE NUMBER OF ROLLS.
670 REM FIND THE AVERAGE OF THE MAXIMUM CAPITOL.
680 A1=0
690 A2=0
700 A3=0
710 FOR K=1 TO 25
720 A1=A1+N(K)
730 A2=A2+R(K)
740 A3=A3+M(K)
750 NEXT K
760 A1=A1/25
770 A2=A2/25
780 A3=A3/25
790 PRINT @Z9:" "
800 PRINT @Z9: USING 810:" AVERAGES  ";A1;"  ";A2;" ";A3
810 IMAGE 11A,5D.2D,3A,5D.2D,2A,5D.2D
820 END
```

TRIAL NUMBER	NUMBER OF PLAYS	NUMBER OF ROLLS	MAXIMUM CAPITAL
1	19	39	6
2	141	525	12
3	1345	4546	60
4	209	694	16
5	487	1703	21
6	43	166	9
7	89	272	14
8	91	301	11
9	9	32	5
10	4289	14629	94
11	31	118	7
12	1229	4079	38
13	11	25	6
14	17	67	6
15	297	1033	26
16	11	33	5
17	9	20	6
18	39	159	7
19	9	19	6
20	1085	3793	30
21	15	51	6
22	35	108	8
23	343	1164	22
24	145	455	17
25	7	25	6
AVERAGES	400.20	1362.24	17.76

Figure 2

A Grade Maintenance Program for the Apple II with Disk Drive

Jim Hunter

One of the most laborious tasks for any teacher is maintaining accurate grade records. This program is designed to minimize the time required for that task and to maximize the accuracy with which it is done. Its features include: several output formats, weighting of individual marks, provision for missing grades and make-up work, and a turnkey approach which allows someone with little or no programming ability to operate it.

The program will be discussed with regard to its construct, its utilization, and its modification for individualized requirements. The accompanying listing of the initialization routine and the actual operations program should be referred to as the discussion proceeds. An added bonus, for those of you who do not have need of a grade keeping system, is that careful scrutiny of these listings will reveal one way in which to set up, maintain, and randomly access data files on the Apple II Disk II system. Please note that both programs are written in Applesoft, and large amounts of data may require large memory sizes in RAM. A firmware (ROM) card for Applesoft is also a big help for operating this routine quickly.

The initialization routine is the first step in running the package. You will note that, during this initialization process, you are called upon to indicate the period number and class name. My application was at the secondary level, but by modifying the file labels, one could have grades by subject matter for an elementary class. This program is to be run one time for each disk (class/subject). After that, the main program will boot automatically with the turnkey menu.

The operator's first choice in the main menu is to "make input." This is used to enter new data or to change old information. The second main menu item is "read output," and this is used to examine or print status reports. To exit the program, choice 3 (terminate work) is used.

Jim Hunter, Byte Shop of Westminster, 14300 Beach Blvd., Westminster, CA 92683.

The Program

Lines 10 to 120 are used to dimension all arrays and to print the main menu. Lines 1000 and 2400 are used to enter new or updated material. Lines 5000 to 7080 call the output routines. The remainder of the listing consists of several subroutines, as follows:

```
10000 GET CLASS DATA
11000 PRINT PAGE HEADINGS
12000 GET LAST ASSIGNMENT #
13000 GET LAST ROSTER #
14000 READ ROSTER NAMES
15000 SORT WEIGHTED AVERAGES
16000 LOAD GRADES INTO RAM
17000 LOAD ASSIGNMENT NAMES
18000 LOAD ASSIGNMENT WEIGHTS
19000 CRT FORMAT
```

The subroutines from 20000 to 28330 are output formatting devices. More about these formats will be discussed later. Lines 31000 to 32000 are used to turn the printer on and off.

Operation

From the main menu, the first branch loads minimal data for manipulation. The second branch, output, loads ALL files for use in the printout. It is suggested that, when running this program, the operator first make any and all inputs prior to making output runs. The input portion allows the operator to change any data, except the "classpoop" information which is generated during the initialization process.

When entering new data, a code is used for a student who did not complete the work, but who will do so at a future time. For this purpose, enter "1" as the grade, and that mark will not be included in any of the averages. Later, when the work is made up, use the "input" routine to alter the mark. It will then automatically be included in the averages for the student, the class, and the assignment.

One of the problems which I encountered when developing this program was the fact that not all assignments are worth the same amount of credit. For this reason, when entering a class set of marks, you will be asked for the assignment number, assignment name, and the

ADVANCED COMPUTER PRODUCTS

NEW CATALOG AVAILABLE NOW!

THE FIRST TO OFFER PRIME PRODUCTS TO THE HOBBYIST AT FAIR PRICES NOW LOWERS PRICES EVEN FURTHER!

1. Proven Quality Factory tested products only, no re-tests or fallouts. Guaranteed money back. We stand behind our products.

1979 CATALOG NOW AVAILABLE.

Send $1.00 for your copy of the most complete catalog of computer products. A must for the serious computer user.

STATIC RAM BOARDS

Now, just more than 1½¢ per bit!

S-100 32K (uses 2114)

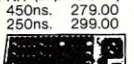
ASSEMBLED		Kit	
450ns.	599.00	450ns.	539.95
250ns.	699.95	250ns.	599.95

Bare Board 49.95
Bare Board w/all parts less mem. 99.95

S-100 16K (uses 2114) KIT (exp. to 32K)

ASSEMBLED		KIT	
450ns.	325.00	450ns.	279.00
250ns.	375.00	250ns.	299.00

Bare Board 49.95



LOGOS I 8K

ASSEMBLED		KIT	
450 ns.	169.95	450ns.	125.95
250ns.	189.95	250ns.	149.95

Bare PC Board w/Data $21.95
Now over 1 year successful field experience
"Special Offer" Buy (4) 8K 450ns. Kits $117.00

FLOPPY DISK DRIVES

1. VISTA V-80 MINIDISK FOR TRS-80
   - ★ 23% More Storage Capacity - 40 Tracks
   - ★ 40 track patch now avail.
   - ★ Faster Drive - Up to 8 Times Faster
   - 2 Drive Cable Add $29.95
   - 4 Drive Cable Add $39.95
   - 395.00
2. VISTA V-200 MINI-FLOPPY SYSTEM
   - ★ 204K Byte Capacity ★ w/CPM, Basic "E"
   - ★ Double Density Drive
   - ★ One Double Density Controller w/Case & P.S.
   - V-200 699.00

Add to your EXIDY, HORIZON, and other S-100 computers.

3. VISTA V-1000 FLOPPY DISK SYSTEM
   - ★ (2) Shugart 8" Floppy Disks
   - ★ Controller Card, Cable, Case & P.S.
   - ★ CPM & Basic "E", Instructions & Manual
   - V-1000 1699.00
4. MPI B51-5¼", 40 tracks 279.00
5. Shugart SA400-5¼", 35 tracks 295.00
6. Siemens/GSI FDD100-8 8" 375.00
7. Shugart 800/801R 8" 495.00
8. PERSCI Model 277 Dual 1195.00
9. WANGO/SIEMENS 5¼" Drive 290.00

EXPANDORAM MEMORY KITS

★ Bank Selectable ★ Uses 4115 or 4116 200 ns.
★ Write Protect ★ Power 8VDC, ±16VDC
★ Phantom ★ Lowest Cost/Bit

Expando 32 Kit (4115)		Expando 64 Kit (4116)	
8K	$158.00	16K	$248.95
16K	$199.00	32K	$369.00
24K	$299.00	48K	$469.00
32K	$349.00	64K	$565.00

IMS STATIC RAM BOARDS

$699.00

★ Memory Mapping ★ Low Power
★ Phantom ★ Assembled & tested
Recommended by Alphamicrosystems

	250 ns.	450 ns.
8K Static	$209.00	$189.00
16K Static	$449.00	$399.00
32K Static	$799.00	$699.00

ANADEX PRINTER

Model DP-8000 compact, impact, parallel or serial. Sprocket feed, 80 cols, 84 lines/min., bi-directional. New only $895.00

VERBATIM™ DISKETTES

LOW PRICE TOP QUALITY DISKETTES

★ 5¼" Minidiskettes ★
Softsector, 10 Sector, 16 Sector
$4.25 Each, 10/39.95
★ 8" Standard Floppy Disks ★
Soft Sector, Hard Sector
$4.50 Each, 10/41.95
*Add 4.95 for 10 Pack in Deluxe Disk Holder

6800 MOTOROLA EXORCISER COMPATIBLE

9600 MPU Module w/6802 CPU $495.00
9601 16 Slot Mother Board 175.00
9602 Card Cage (19" Retma Rack Mount) ... 75.00
9603 8 Slot Mother Board 100.00
9604 Switchmode System Power Supply 250.00
9610 Utility Prototyping Board 36.00
9616 Quad 8K Eprom Module .... ***
9620 16 Channel Parallel I/O Module 295.00
9622 Serial/Parallel I/O Combo .... ***
9626 8K Static RAM Module 295.00
9627 16K Static 450ns 495.00
9630 Card Extender 68.00
9640 Multiple Programmable Timer (24 Timers) 395.00
9650 8 Channel Duplex Serial I/O 395.00
96103 32/32 I/O Module 275.00
96702 32 Point Reed Relay Module 350.00

6800 BARE BOARDS

9620-0	$45.00	9603-0	27.00
9626-0	45.00	9600	55.00
9650-0	45.00	96103	55.00
9601-0	50.00	96702	55.00

Also AMI EVK System in Stock

APPLE/EXIDY/EXPANDO TRS 80 16K-UPGRADE KIT

★ 16K with Jumpers & Instructions for either Level I or Level II $74.95
★ 16K for Apple II Upgrade $74.95
Special: TRS80 Schematic .... $ 4.95
Expansion Interface Schematic .... $ 4.95

TRS 80 TO S-100 PET TO S-100 ADAPTER

Allows Pet/TRS 80 to be interfaced to popular S-100 Bus.
Pet to S-100 Kit $189.95
Assembled $269.95
TRS 80 to S-100 HUH 8100 Kit $275.00
Assembled $355.00

KEYBOARD ASCII ENCODED

One time purchase of NEW Surplus keyboards. From the Singer Corporation. The keyboard features 128 ASCII characters in a 63 key format, MOS encoder circuitry "N" key rollover, lighted shift lock, control, escape and repeat functions. Ltd Qty **63 KEY $59.95**

UV "Eprom" Eraser

Model UVs-11E $69.95
Holds 4 Eprom's at a time. Backed by 45 years experience.
Model S-52T ... $265.00
Professional Industrial Model

TARBELL FLOPPY INTERFACE

★ Z80/8080 S 100 Compatible ★ Uses CPM
Assembled for Shugart .... **SALE $229.00**
Assembled Other Drives $269.95
Kit $179.95
Bare Board .... $36.95 (Doc. Add $10.00)
Note: For CPM Add $70.00. Documentation Add $20.00
Vista Double Density 5¼" Controller Assem. $299.00
SD Versa Floppy Kit $159.95
SD Versa Floppy Assembled $189.95
Tarbel Cassette I/O Kit $115.00
Sale ★ 1771-01 Floppy Chip $27.95

BYTE USER 8K EPROM BOARD

★ Power on Jump ★ Reset Jump
Assembled & Tested $94.95
Byteuser Kit $64.95
Bare PC Board $21.95
Special Offer: Buy 4 kits only $59.95 each
MR-8 8K w/1K Ram $99.50
MR-16 16K w/1K Ram $99.50
EPM-1 4K 1702 .... $59.95
EPM-2 2708 or 2716 Eprom $69.95

Z-80/Z-80A/8080 CPU BOARD

★ On board 2708 ★ 2708 included (450ns.)
★ Power on jump ★ completely socketed
Assembled and tested $185.00
Kit $129.95
Bare PC Board .... $ 34.95
★ For 4MHz Speed Add $15.00
8080A Kit .... $ 99.95
8080A Assembled $149.95

S-100 MOTHERBOARD SPECIAL

8 slot expandable w/9 conn.
reg $69.95 .... NOW $52.95

TARBELL FLOPPY CONTROLLER

Card assembled and tested for use with Shugart Drives $ SALE PRICE only $229.00

ACOUSTIC MODEM

NOVATION CAT™
0-300 Baud
Bell 103
Answer, Originate $198.00

ACOUSTIC COUPLER SPECIAL

AJ MODEL A30
SPECIAL PURCHASE
OF SURPLUS UNITS
AVAILABILITY LIMITED $29.95

DATA BOOKS • COMPUTER BOOKS

1979 IC Master	49.95	Intel MCS 80 Manual	7.95
NSC TTL Data	3.95	Intel MCS 40 Manual	4.95
NSC Linear	4.95	AMD 8080A Manual	5.95
NSC Linear App Notes II	3.95	AMD Schottky Databook	4.95
NSC CMOS	3.95	AMI MOS/LSI Data	3.95
NSC Memory	3.95	GI MOS/LSI Data	4.95
Intel Databook	4.95	Harris Analog Databook	4.95
Intel MCS 85 Manual	7.50	TI Linear Control Data	3.95

SALE • OSBORNE BOOKS • SALE

	Reg.	Sale
Intro to Micros Vol. 0	~~8.50~~	7.75
Intro to Micros Vol. I	~~8.50~~	7.75
8080A Programming	~~8.50~~	7.75
6800 Programming	~~8.50~~	7.75
Z80 Programming	~~8.50~~	7.75
Vol. II Some Real Microprocessors w/Binder	~~30.00~~	27.50
Vol. III Some Real Support Devices w/Binder	~~20.00~~	18.50
Intro to Micros Vol. III	~~20.00~~	18.50

SALE • DILITHIUM COMPUTER BOOKS • SALE

Understanding Computers	~~8.95~~	7.95
8080Microcomputer Experiments	~~13.95~~	11.95
Beginning BASIC	~~9.95~~	8.95
Beginners Glossary & Guide	~~6.95~~	5.95
Peanut Butter & Jelly Guide to Computers	~~7.95~~	6.95
8080 Machine Language Programming	~~7.95~~	6.95
Home Computers Vol. I Hardware	~~7.95~~	6.95
Home Computers Vol. II Software	~~13.95~~	11.95
Starship Simulator	~~7.95~~	6.95

MICROPROCESSORS

Z8000 .... $ CALL
Z-80 .... 14.95
Z-80A 16.95
F-8 (3850) 16.95
2650 .... 18.95
CD1802 .... 17.95
8080A 9.95
8080A-4MHz 19.95
SALE 8085 .... 19.95
8008-1 .... 14.95
2901 .... 12.95
2901A 19.95
TMS 9900JL 49.95
CP1600 .... 39.95
6502 .... 11.50
6502A 19.95
IM6100 .... 29.95
6800 .... 13.95
6802P 24.95
8035 .... 19.95
8755 .... 49.95
8748 .... 69.95
6809 .... $ CALL
8086 .... $ CALL

SUPPORT DEVICES

AM9511 Arith. Processor $195.00
AM 9511-1 300 ns 245.00
AM9517 DMA Controller 71.95
AM9519 Universal Interrupt 24.95
3881 (Z80 PIO) 9.95
3881-4 (4MHz) 14.95
3882 (Z-80 CTC) 9.95
3882-4 (4MHz) 14.95
3883 SIO 29.95
3884 DMA 49.95
8205/74S138 Decoder 2.95
8212 8 bit I/O 2.50
8214 Priority Int. 4.50
8216 Bus Driver 2.50
8224 Clock Gen. 2.95
8224-4 (4MHz) 9.75
8226 Bus Driver 3.95
8T26 Bus Driver 2.39
8228 Sys. Control 5.95
8238 Sys. Cont. 6.25
8251 Prog. I/O 6.95
8253 Int. Timer 19.50
8255 Prog. I/O 5.95
8257 Prog. DMA 17.95
8259 Prog. Int. 19.95
8275 CRT Controller 49.95
8279 Prog. Keyboard 16.95
6810-1 128 x 8 RAM 4.75
6820 PIA 5.95
6821 PIA 6.50
6828 Priority Int. 8.95
6834-1 512 x 8 Eprom 12.95
6850 ACIA 5.95
6852 Serial Adapter 8.95
6845/HD46505CRT Contr. 39.95
6860 Modem 8.95
6862 Modulator 11.95
6871A 1.0MHz OSC 25.95
6875 .... 8.25
6880 Bus Driver 2.39
MC68488 .... 19.95
1821 SCD 1K RAM 25.00
1822 SCD 256 x 4 RAM 16.95
1824 CD 32 x 8 RAM 9.95
1852 CD 8 bit I/O 10.95
1854 Uart 10.95
1856 CD I/O 8.95
1857 CD I/O 8.95
6520 PIA 7.50
6522 Mult. 9.25
6530-002 .... 15.50
6530-003 .... 15.50
6530-004 .... 15.50
6530-005 .... 15.50

DYNAMIC RAMS

416/4116 16K (16 Pin) 9.95
Set of 8 416's 74.95
4115 8K (16 Pin) 6.95
4050 4K x 1 (18 Pin) 4.25
4060 4K x 1 (22 Pin) 4.95
4096 4K x 1 (16 Pin) 3.95
2104 4K x 1 (16 Pin) 4.75
4027 4K x 1 (16 Pin) 4.95

5261	1.95	1103	1.95
5262	1.95	4008L	4.95
5270	4.95	6605	7.95
5280	4.95	6604	4.95
5290	12.45	6002	1.50

PROMS

2708 .... 9.95
2708-6 .... 7.50
1702A 3.95
2732 .... 99.00
2716-5V 44.95
2716-5V, 12V 29.95
2758 5V 29.95
5203AQ 11.95
5204AQ 9.95
6834-1 .... 12.95
IM 5610 .... 2.95
SALE 8223 32 x 8 .... 2.95
82S115 512 x 8 (TS) 16.95
82S123 32 x 8 .... 2.50
82S126 256 x 4 .... 3.50
82S129 256 x 4 (TS) 3.50
82S130 512 x 4 (OC) 6.50
NSC DM7578 32 x 8 .... 2.95

CHARACTER GEN

2513-001 (5V) Upper 9.50
2513-005 (5V) Lower 10.95
2513-ADM3 (5V) Lower 14.95
MCM6571 .... 10.75
MCM6571A 10.75
MCM6574 .... 14.50
MCM6575 .... 14.50

UARTS/USRTS

TR1602B (5V, 12V) 3.95
AY51013 (5V, 12V) 4.95
AY51014A/1612 (5-14V) 6.95
AY51013A/1863 (5V) 6.95
TMS 6011 (5V, 12V) 5.50
IM6402 .... 7.95
IM6403 .... 8.95
2350 USRT 9.95
1671B Astros 24.95
SALE TR1472B 9.95

BAUD RATE GEN

MC14411 .... 11.95
4702 .... 14.95
WD1941 .... 9.95

KEYBOARD ENCODERS

AY5-2376 .... 13.75
AY5-3600 .... 13.75
HD0165 .... 9.95
74C922 .... 9.95
74C923 .... 9.95

A/D CONVERTERS

8700 8 bit Binary 13.50
8701 10 bit Binary 22.00
8703 8 bit TS 13.50
9400 Volt to Freq. Conv. 7.25
8750 3-1/2 Digit BCD 13.95
1408L6 6 bit 3.95
1408L8 8 bit 5.95
DACOI A/D 5.95

STATIC RAM HEADQUARTERS

	1-24	25-99	100
21L02 450ns.	1.30	1.25	1.15
21L02 250ns.	1.59	1.55	1.45
2102	1.25	1.15	1.10
2111	3.75	3.65	3.55
2112-1	2.95	2.85	2.65
2101-1	2.90	2.70	2.55
2114L-250ns.	12.95	10.95	9.95
2114L-300ns.	8.95	7.95	6.45
2114L-450ns.	**7.50**	**6.75**	**4.75**
4044/4041 300ns.	9.95	8.75	7.95
4044/4041 450ns.	**7.50**	**6.75**	**4.75**
EMM4200A	9.75	8.75	7.95
EMM4402	7.95	7.25	6.25
EMM4804	12.50	11.50	9.95
5101C-E	7.95	7.95	7.25
upd410 (4200)	10.95	10.25	9.25
AMD9140/41	10.95	10.25	9.25
AMD9130/31	12.95	11.95	10.25
FSC 460/4546 16K CCD Only $18.95 Each			
1101	1.95	1.75	1.25
P2125/93425 (45 ns.)	7.95	7.35	7.25
6508 1K x 1 CMOS	7.95	7.95	7.25
6518 1K x 1 CMOS	7.95	7.95	7.25
74S189 64 bit Ram	3.95	3.25	2.50
8155 I/O w/Ram 21.95			
2147 Low Power 4K Static 14.95 ea.			

CHARGE COUPLED DEVICES

16K CCD - First time offered Fairchild 460 CCD 16K Memory (now you can experiment with CCD technology at a reasonable price. 17 page Application note supplied with each order. Quantity limited)

$18.95 each (reg. 43.00)

CRYSTALS

Microprocessor Timebases TV Game

Frequency	Price	Frequency	Price
1.0MHz	$5.85	6.0MHz	$ 4.95
1.8432	4.95	6.144	4.95
2.0MHz	5.85	6.5536	4.95
2.01MHz	2.95	10.0MHz	4.95
2.097152MHz	5.85	13.0MHz	4.95
2.4576MHz	5.85	14.31818	4.95
3.579545MHz	1.50	18.0MHz	4.95
4.0MHz	4.95	18.432MHz	5.95
4.194304MHz	5.95	20.0MHz	4.95
4.91520MHz	5.95	22.1184MHz	5.95
5.0MHz	4.95	27.0MHz	5.95
5.0688	4.95	36.0MHz	5.95
5.7143MHz	5.95	48.0MHz	5.95
		100KC	12.95

DISPLAYS/OPTO/LED'S

★ 7 SEGMENT ★ CALC ★ CLOCKS ★
DL 704 (CC), DL 707 (CA) .300" Red .... .99
FND 357 (CC) .357" Red .... .99
FND 500/503 (CC) .500" Red .... .99
FND 507/510 (CA) .500" Red .... .99
FND 800/803 (CC) .800" Red 1.75
FND 807/810 (CA) .800" Red 1.75
XAN 3062 .500" Green 1.15
HP5082-7731 (CA) .300" Red .... .99
9 Digit Bubble Mini Calc. Display .... .99
9 Digit Panaplex Display .400" .... .99
9 Digit Fluorescent .300" .... .99
MA1003 12V Auto Clock Module **15.95**
Bezel for MA1003 w/Red Filter 4.95
MA1002A LED 12 hr. Clock Module 10.95
★ HEX DISPLAYS ★ ENCODED DISPLAYS ★
HP 5082-7340 Red Hexidecimal 15.95
HP 5082-7300 Red Nymeric 14.95
TIL 306 Numeric w/Logic 8.95
TIL 308 Number w/Logic 8.95
TIL 309 Number w/Logic 8.95
TIL 311 Hexadecimal 12.95
MAN 2A .320" Red Alpha-Numeric 5.95
MAN 10A .270" Red Alpha-Numeric 8.95

SOCKETS

8 Pin W/W	32	8 Pin S/T	17
14 Pin W/W	37	14 Pin S/T	20
16 Pin W/W	38	16 Pin S/T	22
18 Pin W/W	60	18 Pin S/T	31
20 Pin W/W	90	20 Pin S/T	34
22 Pin W/W	93	22 Pin S/T	35
24 Pin W/W	85	24 Pin S/T	41
28 Pin W/W	1.15	28 Pin S/T	49
40 Pin W/W	1.49	40 Pin S/T	63

TEXTOOL ZERO INSERTION FORCE SOCKETS

16 Pin $ 5.50 24 Pin $7.50
40 Pin $10.25

CONNECTORS

DB25P (RS232) 3.25
DB25S Female 3.75
Hood 1.25
Set w/Hood, Sale $6.50
22/44 W/W, S/T, KIM 2.95
43/86 W/W, S/T, MOT 6.50
50/100 S-100 Connector w/w 4.25
50/100 S-100 Connector s/t 3.25

CTS DIPSWITCHES

CTS206-4 $1.75 CTS206-8 $1.95
CTS206-5 $1.75 CTS206-9 $1.95
CTS206-6 $1.75 CTS206-10 $1.95
CTS206-7 $1.75

NAKED PC BOARD SALE

Z-80 CPU (Ithaca) $34.95
8080A CPU 34.95
8K Static RAM (Logos) 21.95
16K Static RAM (2114) 29.95
32K Static RAM (2114) 49.95
Floppy I/O (Tarbell) 39.95
Cassette I/O (Tarbell) 29.95
8K Eprom (2708) 21.95
1702 Eprom Board 30.00
2708/2716 Eprom (Ithaca) 34.95
2708/2716 Eprom(WMC) 30.00
Realtime Clock 34.95
ACP Proto Bd. (3M Conn.) 27.95
Vector 8800 Proto 19.95
Vector 8803 11 slot MB 29.95
ACP Extender w/Conn 15.95
Video Interface (SSM) 27.95
Parallel Interface (SSM) 27.95
13 Slot MotherBoard (WMC) 32.95
9 Slot MotherBoard (WMC) 29.95
8 Slot MotherBoard (expandable) 34.95

WAVEFORM GENERATORS

8038 Function Gen. 3.95
MC4024 VCO 2.45
LM566 VCO 1.75
XR2206 Function Generator 5.25

FLOPPY DISK I/O

1771-01 8" & Minifloppy 27.95
uPd372 Nec Floppy 49.95
1781 Dual Floppy 29.95
1791 Dual Floppy 39.95

TV INTERFACES

Pixie-Verter 8.50
TV-1 Video Interface 8.95
Microverter 35.00
M&R Modulator 35.00

SPECIAL PURCHASE (while supply lasts)

21L02-4 (450 ns) 100 @ 99¢ ea.
21L02-2 (250 ns) 100 @ $1.15 ea.
TMS4060 NL 4K Dynamic RAMS (pullouts) $1.95 ea.
1488 Line Receiver 100 @ 75¢ ea.
1489 Line Driver 100 @ 75¢ ea.
1489 House Marked 100 @ 50¢ ea.
1496 L Demodulator 25 @ 75¢ ea.

COMPUTER SPECIALS

	LIST	SALE
Apple II Plus w/16K	1195.	990.
PET 2001-16N	995.	895.
Exidy Sorcerer w/8K	895.	795.
Compucolor II w/8K	1495.	1395.
Cromemco Sys III	5990.	4990.
Horizon I w/16K	1599.	1349.
TEI PI208 w/32K dual floppy & CRT (1 avail.)	4995	2995
Pascal Microengine	2995	2395
IPSI 1620 Diable RO	3295.	2695.
Anadex DP 8000	995.	895.
Centronics Micro P-1	595.	395.
Centronics Micro S-1	595.	525.
Soroc IQ 120	995.	850.
Teletype Model 43	1349.	1150.
HiPlot Plotter	1085.	899.
HiPlot Digitizer	795.	735.
Intertube II	895.	784.

★ LED's ★ OPTOISOLATERS ★

LEDS Red, Yellow, Green .185 .... 5/1.00
MCT 2 Photo XSTR HFE 250, 30V .... .99
4N25 Photo XSTR HFE 250, 30V 1.29
4N33 Photo Darlington 1.75
FPT 110B Photo XSTR Flat Lense .... **SALE** 4/1.00

MONTHLY IC SPECIALS

LF13508 JFET Anlog Multi 8 bit 8.95
ICM7208 Seven Decade Counter 17.95
ICM7207 Oscillator Controller 6.95
ICM7045 Precision Sto/Watch Timer 22.95
ICL7107 3½ Digit A/D (LED) 14.95
ICL8211 Voltage Reference 1.95
LM390 Battery OP. Audio Amp 3/1.00
LM1850 Ground Fault IC 3/1.00
LM1800 Phase Lock Loop FM Stereo 3/1.00
LM1820 AM Radio 3/1.00
DS3625 Dual Mos Sense Amp 2.50

1408L8	4/19.95	10147 ECL Ram	9.95
1488/1489	2/1.99	LF356HBiFet	3/1.99
22 Pin S/T Socket	10/1.00	MCM14505	8.95
8223 Prom	2.95	74S89	3/1.99
MK5014 Calc.	2/1.99	74107N	6/1.99
74141N	3/1.99	75452N	8/1.99
8T26/8T28	2.39	741N-14	10/1.99
95H90	9.95	555CN	5/1.99
8281	.50	556CN	3/1.99

TV CHIPS/SOUND

AY38500-1 6 Games B/W $4.95
AY38515 Color Converter 2.95
AY38603-1 Roadrace Game 8.95
AY38605-1 Warfare Game 9.50
AY38606-1 Wipeout Game 9.50
AY38607-1 Shooting Gallery 8.95
AY38910 Gimini Cricket Sound Generator 12.95
SN76477 TI Sound Generator 3.95
MM5320/21 TV Synch Gen. 9.95
MM5369 Prescaler 3.95
LM1889 RF Modulator 3.95
MM571000 NSC Color TV Game 6.95
MM57104 Clock Gen. 3.75
RF Modulator w/Audio 8.95

All Shipments FCM or UPS. Orders under $100.00 add 5% handling and postage. Orders over $100.00 add 2.5% handling & postage. Mastercharge/ B of A /American Express/ COD accepted w/25% deposit. California Residents add 6% tax. Foreign Orders add 8% handling. All parts prime factory tested guaranteed. Add .35 cents for Data.

Retail pricing may vary from Mail Order Pricing. All pricing subject to change without notice.

P. O. BOX 17329 Irvine, California 92713 Phone (714) 558-8813 TWX: 910-595-1565

Retail Store Open Mon. — Sat. Located at 1310 "B" E. Edinger, Santa Ana, CA 92705

CIRCLE 103 ON READER SERVICE CARD

weighting. I used factors from .01 to 1, but the operator can choose anything which he feels comfortable with. A weighting of "1" is suggested for major assignments or exams, while ".1" might be used for quizzes or homework.

Once all new data and updates have been posted, the next logical step is to examine or print output. The output formats number four:

1. A STUDENT FILE.
2. CLASS AVERAGES BY STUDENT.
3. CLASS AVERAGES BY ASSIGNMENT.
4. RANKED AVERAGES BY STUDENT.

The first of these formats is used to evaluate the progress of an individual student. The second gives a class summary by roster number of each student's present average. My experience revealed that weekly posting of this output served two purposes. First, the students got constant feedback on their progress; and second, arguments as to what grade was being earned at progress report time were reduced to zero.

The third type of printout is the class averages by assignment. This is a useful tool to evaluate comparative difficulty of assignments. This output is primarily an instructor's tool. Finally, printout number four is a ranked output from highest to lowest. This also was posted, but I recommend that you check with your students first on that one.

Sample outputs for each format are shown with the listings. The class shown is, of course, fictitious.

Suggestions For Modifications

The most obvious area where modifications would be called for is hard copy output. The printer I was using was hooked into slot 1 with a Parallel Card from Apple. You may want to modify this portion as required. One section of the program as I use it which has been omitted is the actual assignment of grades by the computer. This is accomplished by inserting a subroutine which determines the letter grade and then changing the output statements to reflect that data.

If one were concerned with student anonymity on the ranked listing, names could be deleted and roster numbers could be used exclusively. These modifications, coupled with those mentioned in the beginning of this article, should make the program a tool which will serve the needs of teachers at all levels of instruction.

Summary

A little time spent weekly on the Apple will reap great benefits for both the teacher and the student. By ending the mystique of grades, and thus clearing away the cloud surrounding them, a teacher can spend more time doing what he is paid for...teaching.

And as for all of you non-teachers (now including myself) read over the listings here, and learn of the mysteries of file maintenance on the Apple. It took me several weeks and calls to Cupertino to arrive at this information. If it helps you, then I have served my purpose well.

Disk copies of this program are available from:

The Byte Shop of Westminster, 14300 Beach Blvd., Westminster, CA 92683.

The cost is $40, with documentation, $30 without. (This article is really all you need). □

SAMPLE PRINT OUTS
Format #1

```
PERIOD 1                ENGLISH LITERATURE
DATE OF RUN: 7 JANUARY 1979
INCLUDES ASSIGNMENTS TO #6
LAST ASSIGNMENT NAME IS:SHAKESPEARE ESSAY

ROSTER NUMBER: 3       CLAIRMONT BEVERLY

    ASSIGNMENT                  STUDENT
NO.  WT.     NAME                 MARK

1   .1   FIRST QUIZ               89
2   .1   SECOND QUIZ              87
3   1    CLASS ESSAY              92
4   .33  PARA DRILL               59
5   1    FIRST EXAM               92
6   .8   SHAKESPEARE ESSAY        100
AVERAGE TO DATE IS: 90.4
```

SAMPLE PRINT OUTS
Format #2

```
PERIOD 1                ENGLISH LITERATURE
DATE OF RUN: 7 JANUARY 1979
INCLUDES ASSIGNMENTS TO #6
LAST ASSIGNMENT NAME IS:SHAKESPEARE ESSAY

*********** CLASS SUMMARY ***********
ROSTER   STUDENT                  CURRENT
NUMBER   NAME                       AVG

1        ABERCROMBY JOHN            89.2
2        BENNET CHRIS               71.6
3        CLAIRMONT BEVERLY          90.4
4        DEMAUPASSANT GUY *         82
5        EVERLY BOB                 85
6        FRITCHMAN CLAUDE           84.6
7        GEORGE GORGEOUS            81
```

```
B.   The * next to Guy's name means that he
     is missing an assignment.
```

SAMPLE PRINT OUTS
Format #3

```
PERIOD 1                ENGLISH LITERATURE
DATE OF RUN: 7 JANUARY 1979
INCLUDES ASSIGNMENTS TO #6
LAST ASSIGNMENT NAME IS:SHAKESPEARE ESSAY

********** ASSIGNMENT SUMMARY *********

   ASSIGNMENT           NUMBER     CLASS
NO.  WT.    NAME        MISSING     AVG.

1    .1   FIRST QUIZ      0          78
2    .1   SECOND QUIZ     0          86
3    1    CLASS ESSAY     0          90
4    .33  PARA DRILL      1          74
5    1    FIRST EXAM      0          95
6    .8   SHAKESPEARE ESSAY0         62

WEIGHTED CLASS AVERAGE TO DATE IS: 82
```

SAMPLE PRINT OUTS
Format #4

```
PERIOD 1                ENGLISH LITERATURE
DATE OF RUN: 7 JANUARY 1979
INCLUDES ASSIGNMENTS TO #6
LAST ASSIGNMENT NAME IS:SHAKESPEARE ESSAY

********* RANKED AVERAGES *********

CLASS     STUDENT                    CURRENT
RANKING   NAME                         AVG

1         CLAIRMONT BEVERLY            90.4
2         ABERCROMBY JOHN              89.2
3         EVERLY BOB                   85
4         FRITCHMAN CLAUDE             84.6
5         DEMAUPASSANT GUY *           82
6         GEORGE GORGEOUS              81
7         BENNET CHRIS                 71.6
```

COME TO BOSTON

FRI. SAT. SUN. **Sept. 28, 29 & 30** FOR THE

NORTHEAST PERSONAL & BUSINESS COMPUTER SHOW

HYNES AUDITORIUM, PRUDENTIAL CENTER, BOSTON

Interested In Personal Computers?

This is the place for you! You'll meet and talk with hundreds of manufacturers, distributors and retailers... all showcasing their new 1980 Micro, Mini and Small Computer Systems.

You'll see them all...Radio Shack, Pet, RCA, Compucolor, Heathkit...you name it! All the major terminal and peripheral companies too, plus software developers magazine editors and book publishers. Yes, it'll be the largest showing of personal computer hardware, software and services ever assembled in the Northeast!

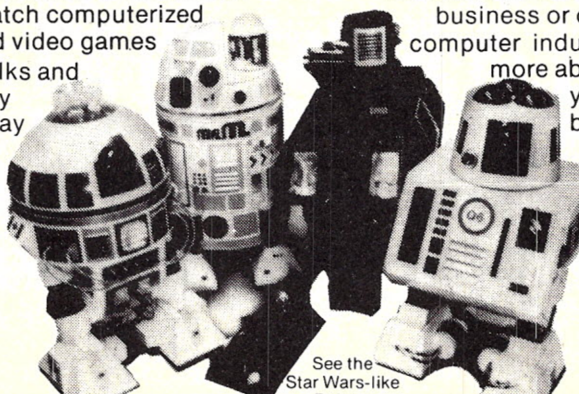
You'll be enthralled, entertained and educated. You'll see computer art, graphics and animation. You'll hear computer synthesized music, watch computerized amusements, play electronic and video games and attend scores of free tech talks and briefings given by internationally recognized speakers. And you may win a free computer given away as a door prize! Don't miss the largest gathering of computers and computerists! No pre-registration necessary. Tickets available at the door. Adult admission $5.00.

Interested In Business Systems?

Interested in Business Systems? This will be your one opportunity to see all of the Mini, Micro and Moderate Sized Computer Systems under one roof. Your attendance at this show is a must if you, or your company, are contemplating the purchase of any type of computer or office equipment.

You'll see them all...the big (and small) names in computers, data and word processing eqiupment, peripherals and software. You'll attend dozens of free, easily-understood briefings on how computers can help you in your business or profession.

So if you're considering a computer or computer-related service...starting your own computer business or changing your job within the computer industry...or if you'd just like to learn more about computers as they relate to your personal life and to your business or profession, come to the show to see the Computerized World of Tomorrow...Today. And bring the entire family, It's a fun-filled educational experience. No pre-registration necessary. Tickets available at the door. Adult admission five dollars.



See the Star Wars-like Robots

Produced by Northeast Expositions, P.O. Box 678, Brookline Village, MA 02147 (617) 522-4467

INITIALIZATION ROUTINE

```
]PR#0
]LIST

10 D$ = ""
20  PRINT "";"NOMON I,O,C"
30  HOME
40  PRINT "WHAT IS THE PERIOD NUM
     BER?"
50  INPUT P
60  PRINT "WHAT IS THE CLASS NAME
     ?"
70  INPUT CLASS$
80  PRINT D$;"OPEN CLASSPOOP$,L25
     "
90  PRINT D$;"WRITE CLASSPOOP$,R1
     "
100  PRINT P
110  PRINT D$;"WRITE CLASSPOOP$,R
     2"
120  PRINT CLASS$
130  PRINT D$;"CLOSE CLASSPOOP$"
140  PRINT D$;"OPEN ASSNAME$,L25"

150  FOR I = 1 TO 40
160  PRINT D$;"WRITE ASSNAME$,R";
     I
170  PRINT " "
180  NEXT I
190  PRINT D$;"CLOSE ASSNAME$"
200  PRINT D$;"OPEN L%,L4"
210  FOR I = 1 TO 2
220  PRINT D$;"WRITE L%,R";I
230  PRINT 0
235  NEXT I
240  PRINT D$;"CLOSE L%"
250  PRINT D$;"OPEN ASSWT%,L4"
260  FOR I = 1 TO 40
270  PRINT D$;"WRITE ASSWT%,R";I
280  PRINT 0
290  PRINT D$;"CLOSE ASSWT%"
300  PRINT D$;"OPEN NAME$,L25"
310  FOR I = 1 TO 40
320  PRINT D$;"WRITE NAME$,R";I
330  PRINT " "
340  NEXT I
350  PRINT D$;"CLOSE NAME$"
```

MAIN PROGRAM LISTING

```
]LIST

10  PRINT "";"NOMON I,O,C"
20 D$ = ""
30  DIM NAME$(40),G(40,50),W(50),ASSNAME$(50)
35  DIM AVG(40)
```

```
2170  PRINT D$;"OPEN G%";A;",L4"
2180  FOR I = 1 TO LN
2190  PRINT D$;"WRITE G%";A;",R";I
2195  PRINT G(I,A)
2200  NEXT I
2210  PRINT D$;"CLOSE G%";A
2220  PRINT D$;"APPEND ASSNAME$,L25"
2230  PRINT D$;"WRITE ASSNAME$,R";A
2240  PRINT ASSNAME$(A)
2250  PRINT D$;"CLOSE ASSNAME$"
2260  PRINT D$;"APPEND ASSWT%,L4"
2270  PRINT D$;"WRITE ASSWT%,R";A
2280  PRINT W(A)
2290  PRINT D$;"CLOSE ASSWT%"
2300  GOSUB 11000
2310  PRINT D$;"OPEN L%,L4"
2320  PRINT D$;"WRITE L%,R2"
2330  PRINT A
2340  PRINT D$;"CLOSE L%"
2350  GOSUB 11000
2360  PRINT "ANOTHER SET OF MARKS";
2370  INPUT C$
2380  IF C$ <  > "Y" AND C$ <  > "YES" THEN 60
2390  GOSUB 12000
2400  GOTO 2010
5000  GOSUB 11000
5010  PRINT
5020  PRINT "WHAT DO YOU WANT TO CHANGE?"
5030  PRINT : PRINT "  1.   STUDENT NAME."
5040  PRINT "  2.   STUDENT GRADE."
5050  PRINT "  3.   ASSIGNMENT NAME."
5060  PRINT "  4.   ASSIGNMENT WEIGHT."
5065  PRINT "  5.   RETURN TO MAIN MENU."
5070  PRINT : PRINT "WHAT DO YOU WISH, MASTER"
5080  INPUT C
5090  ON C GOTO 5100,5300,5500,5700,60
5100  PRINT "WHAT ROSTER NUMBER DO YOU WISH?";
5110  INPUT N
5120  PRINT D$;"OPEN NAME$,L25"
5130  PRINT D$;"READ NAME$,R";N
5140  INPUT NAME$(N)
5150  PRINT D$;"CLOSE NAME$"
5160  PRINT "NAME NUMBER ";N;" IS NOW ";NAME$(N)
5170  PRINT "WHAT IS THE NEW NAME?";
5180  INPUT NAME$(N)
5185  IF NAME$(N) = "AVAILABLE" THEN  GOSUB 7000
5190  PRINT D$;"APPEND NAME$,L25"
5200  PRINT D$;"WRITE NAME$,R";N
5210  PRINT NAME$(N)
5230  PRINT D$;"CLOSE NAME$"
5240  PRINT : PRINT "DO YOU WANT TO CHANGE ANOTHER NAME?";
5250  INPUT C$
5260  IF C$ <  > "Y" AND C$ <  > "YES" THEN 5000
5270  GOTO 5100
5300  GOSUB 11000
5310  PRINT : PRINT "WHAT ROSTER NUMBERT DO YOU WANT";
5320  INPUT N
5330  PRINT : PRINT "WHAT ASSIGNMENT NUMBER DO YOU WANT";
5340  INPUT A
5350  PRINT D$;"APPEND G%";A;",L4"
5360  PRINT D$;"READ G%";A;",R";N
5370  INPUT G(N,A)
5380  PRINT D$;"CLOSE G%";A
5390  PRINT : PRINT "THE CURRENT GRADE FOR ROSTER NUMBER ";N
5400  PRINT "FOR ASSIGNMENT NUMBER ";A;" IS ";G(N,A)
5410  PRINT : PRINT "WHAT IS THE NEW GRADE";
```

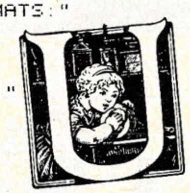
```
36  DIM M(40)
37  DIM MA(50)
38  DIM X(40)
39  DIM ASSTAVG(50)
40  HOME
50  GOSUB 10000
60  GOSUB 11000
65  PRINT
70  PRINT "YOU HAVE THREE INITIAL OPTIONS:"
75 Q = 0
80  PRINT : PRINT "  1.   MAKE INPUT. "
90  PRINT "  2.   READ OUTPUT. "
100  PRINT "  3.   TERMINATE WORK. "
110  PRINT : INPUT "WHAT IS YOUR PLEASURE";C
120  ON C GOTO 1000,20000,32000
1000  HOME
1010  GOSUB 11000
1020  PRINT : PRINT "NEXT CHOICE:"
1030  PRINT "  1.   ENTER NEW DATA. "
1040  PRINT "  2.   CHANGE EXISTING DATA. "
1050  PRINT : INPUT "WHAT IS YOUR PLEASURE";C
1060  ON C GOTO 1100,5000
1100  GOSUB 11000
1110  PRINT : PRINT "WHICH OF THE FOLLOWING DO YOU WISH:"
1120  PRINT : PRINT "  1.   ENTER CLASS ROSTER. "
1130  PRINT "  2.   ENTER CLASS SET OF MARKS. "
1150  PRINT : INPUT "PICK ONE:";C
1160  ON C GOTO 1200,2000
1200  GOSUB 11000: PRINT
1210  PRINT "AS EACH NUMBER APPEARS, ENTER THE"
1220  PRINT "CORRESPONDING ROSTER NAME.  WHEN YOU"
1230  PRINT "HAVE EXHAUSTED THE NAMES, THEN TYPE"
1240  PRINT " <LAST>. "
1250  FOR I = 1 TO 40
1260  PRINT I;
1270  INPUT NAME$(I)
1280  IF NAME$(I) = "LAST" THEN LN = I - 1
1290  IF NAME$(I) = "LAST" THEN I = 40
1300  NEXT I
1310  PRINT D$;"OPEN NAME$,L25"
1320  FOR I = 1 TO LN
1330  PRINT D$;"WRITE NAME$,R";I
1340  PRINT NAME$(I)
1350  NEXT I
1360  PRINT D$;"CLOSE NAME$"
1370  PRINT D$;"OPEN L%,L4"
1380  PRINT D$;"WRITE L%,R1"
1390  PRINT LN
1400  PRINT D$;"CLOSE L%"
1410  HOME
1420  GOTO 60
2000  GOSUB 11000: GOSUB 12000: GOSUB 13000: GOSUB 14000
2010  PRINT
2020  PRINT "THE LAST ASSIGNMENT NUMBER WAS: ";LA
2030  PRINT "WHAT IS THIS ASSIGNMENT NUMBER"
2040  INPUT A
2050  GOSUB 11000
2060  PRINT "WHAT IS THE ASSIGNMENT NAME"
2070  INPUT ASSNAME$(A)
2080  PRINT "WHAT IS THE WEIGHTING FOR THIS WORK"
2090  INPUT W(A)
2100  GOSUB 11000
2110  PRINT : PRINT "AFTER THE ROSTER NUMBER AND NAME, YOU"
2120  PRINT "ARE TO ENTER THE SCORE FOR THAT STUDENT"
2130  FOR I = 1 TO LN
2140  PRINT I; TAB( 5)NAME$(I);
2150  INPUT G(I,A)
2160  NEXT I
5420  INPUT G(N,A)
5430  PRINT D$;"APPEND G%";A;",L4"
5440  PRINT D$;"WRITE G%";A;",R";N
5450  PRINT G(N,A)
5460  PRINT D$;"CLOSE G%";A
5470  PRINT : PRINT "DO YOU WANT TO CHANGE ANOTHER GRADE";
5480  INPUT C$
5485  IF C$ <  > "Y" AND C$ <  > "YES" THEN  GOTO 5000
5490  GOTO 5300
5500  GOSUB 11000
5510  PRINT : PRINT "WHAT ASSIGNMENT DO YOU WANT TO CHANGE";
5520  INPUT A
5530  PRINT D$;"APPEND ASSNAME$,L25"
5540  PRINT D$;"READ ASSNAME$,R";A
5550  INPUT ASSNAME$(A)
5560  PRINT D$;"CLOSE ASSNAME$"
5570  PRINT : PRINT "THE CURRENT NAME FOR ASSIGNMENT NO.  ";A
5580  PRINT "IS ";ASSNAME$(A)
5590  PRINT : PRINT "WHAT IS THE NEW ASSIGNMENT NAME";
5600  INPUT ASSNAME$(A)
5610  PRINT D$;"APPEND ASSNAME$,L25"
5620  PRINT D$;"WRITE ASSNAME$,R";A
5630  PRINT ASSNAME$(A)
5640  PRINT D$;"CLOSE ASSNAME$"
5650  PRINT : PRINT "DO YOU WANT TO CHANGE ANOTHER NAME";
5660  INPUT C$
5670  IF C$ <  > "Y" AND C$ <  > "YES" THEN 5000
5680  GOTO 5500
5700  GOSUB 11000
5710  PRINT : PRINT "WHAT ASSIGNMENT DO YOU WANT TO CHANGE"
5720  PRINT "THE WEIGHTING ON";
5730  INPUT A
5740  PRINT D$;"APPEND ASSWT%,L4"
5750  PRINT D$;"READ ASSWT%,R";A
5760  INPUT W(A)
5770  PRINT D$;"CLOSE ASSWT%"
5780  PRINT : PRINT "THE CURRENT WEIGHTING FOR ASSIGNMENT "
5785  PRINT "NUMBER ";A;" IS ";W(A)
5790  PRINT : PRINT "WHAT IS THE NEW WEIGHTING";
5800  INPUT W(A)
5810  PRINT D$;"APPEND ASSWT%,L4"
5820  PRINT D$;"WRITE ASSWT%,R";A
5830  PRINT W(A)
5840  PRINT D$;"CLOSE ASSWT%"
5850  PRINT : PRINT "DO YOU WANT TO CHANGE ANOTHER WEIGHT";
5860  INPUT C$
5870  IF C$ <  > "Y" AND C$ <  > "YES" THEN 5000
5880  GOTO 5700
7000  GOSUB 11000
7010  GOSUB 12000
7020  FOR A = 1 TO LA
7030  PRINT D$;"APPEND G%";A;",L4"
7040  PRINT D$;"WRITE G%";A;",R";N
7050  PRINT 0
7060  PRINT D$;"CLOSE G%";A
7070  NEXT A
7080  RETURN
9999  END
10000  REM  SUBROUTINE TO GET CLASS DATA FOR SHEET HEADINGS
10005  PRINT D$;"OPEN CLASSPOOP$,L25"
10010  FOR I = 1 TO 2
10030  PRINT D$;"READ CLASSPOOP$,R";I
10040  INPUT CLASSPOOP$(I)
10050  NEXT I
10060  PRINT D$;"CLOSE CLASSPOOP$"
10070  REM  CLASSPOOP$(1)=PERIOD
10080  REM  CLASSPOOP$(2)=CLASS$
```




```
10090 P$ = CLASSPOOP$(1)
10100 CLASS$ = CLASSPOOP$(2)
10110  RETURN
11000  REM  SUBROUTINE TO PRINT PAGE HEADINGS
11005  HOME
11010  PRINT "PERIOD ";P$;
11020  PRINT  TAB( 40 -  LEN (CLASS$))CLASS$
11030  IF Q = 0 THEN  RETURN
11040  PRINT "DATE OF RUN: ";TIME$
11050  PRINT "INCLUDES ASSIGNMENTS TO #";LA
11060  PRINT "LAST ASSIGNMENT NAME IS:";ASSNAME$(LA)
11070  RETURN
12000  REM  SUBROUTINE TO GET LAST ASSIGNMENT NUMBER
12005  PRINT D$;"OPEN L%,L4"
12010  PRINT D$;"READ L%,R2"
12020  INPUT LA
12030  PRINT D$;"CLOSE L%"
12040  RETURN
13000  REM  SUBROUTINE TO GET LAST ROSTER NUMBER
13005  PRINT D$;"OPEN L%,L4"
13010  PRINT D$;"READ L%,R1"
13020  INPUT LN
13030  PRINT D$;"CLOSE L%"
13040  RETURN
14000  REM  SUBROUTINE TO READ ROSTER NAMES
14005  PRINT D$;"OPEN NAME$,L25"
14010  FOR I = 1 TO LN
14020  PRINT D$;"READ NAME$,R";I
14030  INPUT NAME$(I)
14040  NEXT I
14050  PRINT D$;"CLOSE NAME$"
14060  RETURN
15000  REM  SUBROUTINE TO SORT WEIGHTED AVERAGES
15005 M = 0
15010 W = 0:T = 0
15020  FOR I = 1 TO LA
15025  IF G(N,I) = 1 THEN M = M + 1
15030  IF G(N,I) = 1 THEN  GOTO 15060
15040 W = W + W(I)
15050 T = T + G(N,I) * W(I)
15060  NEXT I
15070 AVG(N) =  INT ((T / (W * 100)) * 1000) / 10
15075 M(N) = M
15080  RETURN
16000  REM  SUBROUTINE TO GET ALL GRADES
16005  FOR A = 1 TO LA
16010  PRINT D$;"OPEN G%";A;",L4"
16020  FOR N = 1 TO LN
16030  PRINT D$;"READ G%";A;",R";N
16040  INPUT G(N,A)
16050  NEXT N
16060  PRINT D$;"CLOSE G%";A
16070  NEXT A
16080  RETURN
17000  REM  GET ASSIGNMENT NAMES
17010  PRINT D$;"OPEN ASSNAME$,L25"
17020  FOR I = 1 TO LA
17030  PRINT D$;"READ ASSNAME$,R";I
17040  INPUT ASSNAME$(I)
17050  NEXT I
17060  PRINT D$;"CLOSE ASSNAME$"
17070  RETURN
18000  REM  GET ASSIGNMENT WEIGHTS
18010  PRINT D$;"OPEN ASSWT%,L4"
18020  FOR I = 1 TO LA
18030  PRINT D$;"READ ASSWT%,R";I
18040  INPUT W(I)
```

```
24095  IF Q = 1 THEN  PR# 1
24100  GOSUB 11000
24110  PRINT : PRINT "************ CLASS SUMMARY ************"
24120  PRINT "ROSTER   STUDENT                    CURRENT"
24130  PRINT "NUMBER   NAME                         AVG. "
24140  PRINT
24150  FOR I = 1 TO LN
24160  PRINT I;
24170  PRINT  TAB( 7)NAME$(I);
24171  PRINT " ";: IF M(I) < 1 THEN 24180
24172  FOR J = 1 TO M(I)
24174  PRINT "*";
24176  NEXT J
24180  PRINT  TAB( 36)AVG(I)
24190  IF I / 13 =  INT (I / 13) THEN  GOSUB 19000
24200  NEXT I
24205  IF Q = 1 THEN  PR# 0
24210  PRINT : PRINT "DO YOU WANT TO SEE THEM AGAIN";
24220  INPUT C$
24230  IF C$ <  > "Y" AND C$ <  > "YES" THEN 20040
24240  GOTO 24100
26000  REM  SUBROUTINE TO CALCULATE CLASS AVERAGES BY ASSIGNMENT,
     AND WEIGHTED CLASS AVERAGE
26005  FOR A = 1 TO LA
26010 T = 0:NT = 0
26015 MA = 0
26020  FOR N = 1 TO LN
26025  IF G(N,A) = 1 THEN MA = MA + 1
26030  IF G(N,A) = 1 THEN  GOTO 26070
26040 T = T + G(N,A)
26050 NT = NT + 1
26060  IF NAME$(N) = "AVAILABLE" THEN NT = NT - 1
26070  NEXT N
26080 ASSTAVG(A) =  INT (T / (NT * 100) * 100)
26085 MA(A) = MA
26090  NEXT A
26095  IF Q = 1 THEN  PR# 1
26100  GOSUB 11000
26110  PRINT : PRINT "********** ASSIGNMENT SUMMARY *********"
26120  PRINT : PRINT "   ASSIGNMENT           NUMBER     CLASS"
26130  PRINT "NO.  WT.     NAME    MISSING       AVG. "
26140  PRINT
26150  FOR A = 1 TO LA
26160  PRINT A;
26170  PRINT  TAB( 6)W(A);
26180  PRINT  TAB( 10)ASSNAME$(A);
26185  PRINT  TAB( 25)MA(A);
26190  PRINT  TAB( 36)ASSTAVG(A)
26200  IF A / 13 =  INT (A / 13) THEN  GOSUB 19000
26210  NEXT A
26211 T = 0
26212 W = 0
26213  FOR I = 1 TO LA
26214 T = T + ASSTAVG(I) * W(I)
26215 W = W + W(I)
26216  NEXT I
26217 CAVG =  INT (T / (W * 100) * 100)
26218  PRINT : PRINT "WEIGHTED CLASS AVERAGE TO DATE IS: ";CAVG
26219  IF Q = 1 THEN  PR# 0
26220  PRINT : PRINT "DO YOU WANT TO SEE THEM AGAIN?";
26230  INPUT C$
26240  IF C$ <  > "Y" AND C$ <  > "YES" THEN  GOTO 20040
26250  GOTO 26100
28000  REM  SUBROUTINE TO OUTPUT RANKED AVERAGES
28005  FOR N = 1 TO LN
28010  GOSUB 15000
28020  NEXT N
```

```
18050  NEXT I
18060  PRINT D$;"CLOSE ASSWTX"
18070  RETURN
19000  REM  SUBROUTINE TO DISPLAY 13 LINES OF OUTPUT FOR CRT USE
     DELETED WITH PRINTER USAGE
19002  IF Q = 1 THEN  RETURN
19005  PRINT : PRINT "RETURN TO CONTINUE"
19010  INPUT C$
19020  HOME
19030  RETURN
20000  INPUT "DO YOU WANT HARD COPY";C$
20002  IF C$ = "Y" OR C$ = "YES" THEN Q = 1
20004  IF C$ = "Y" OR C$ = "YES" THEN  PRINT "TURN ON PRINTER NOW!
     "
20005  INPUT "WHAT IS THE DATE";TIME$
20010  GOSUB 12000: GOSUB 13000
20020  GOSUB 14000: GOSUB 16000
20030  GOSUB 17000: GOSUB 18000
20040  GOSUB 11000
20050  PRINT : PRINT "DISPLAY OF DATA IS IN FOUR FORMATS:"
20060  PRINT "  1.   STUDENT FILE."
20070  PRINT "  2.   CLASS AVERAGES BY STUDENT."
20080  PRINT "  3.   CLASS AVERAGES BY ASSIGNMENT."
20090  PRINT "  4.   RANKED CLASS AVERAGES BY STUDENT."
20095  PRINT "  5.   RETURN TO BASIC MENU."
20097  PRINT "  6.   PRINTER ON."
20098  PRINT "  7.   PRINTER OFF."
20100  PRINT : PRINT "PICK ONE.";
20110  INPUT C
20120  ON C GOTO 22000,24000,26000,28000,60,31000,31100
22000  REM  SUBROUTINE TO OUTPUT INDIVIDUAL STUDENT MARKS WITH AVE
     RAGE TO DATE
22005  GOSUB 11000
22010  PRINT : PRINT "WHAT ROSTER NUMBER DO YOU WANT TO SEE."
22020  INPUT N
22025  IF Q = 1 THEN  PR# 1
22030  GOSUB 11000
22040  PRINT : PRINT "ROSTER NUMBER: ";N;
22050  PRINT  TAB( 40 -  LEN (NAME$(N)))NAME$(N)
22060  PRINT : PRINT "      ASSIGNMENT                  STUDENT"
22070  PRINT "NO.  WT.      NAME                    MARK"
22080  PRINT
22090  FOR A = 1 TO LA
22100  PRINT A; TAB( 5)W(A); TAB( 9)ASSNAME$(A);
22102  IF G(N,A) <  > 1 THEN 22106
22104  PRINT  TAB( 33)"MISSING"
22105  GOTO 22115
22106  PRINT  TAB( 37)G(N,A)
22115  IF A / 13 =  INT (A / 13) THEN  GOSUB 19000
22118  NEXT A
22120  GOSUB 15000
22130  VTAB (20)
22140  PRINT "AVERAGE TO DATE IS: ";AVG(N)
22141  IF Q = 1 THEN  PR# 0
22142  PRINT "DO YOU WANT TO SEE THEM AGAIN";
22144  INPUT C$
22146  IF C$ <  > "Y" AND C$ <  > "YES" THEN  GOTO 20040
22148  GOTO 22030
22150  PRINT : PRINT "CHECK ANOTHER?"
22160  INPUT C$
22170  IF C$ <  > "Y" AND C$ <  > "YES" THEN  GOTO 20040
22180  GOTO 22000
24000  REM  SUBROUTINE TO CALCULATE AVERAGES, PRESENTED BY ROSTER
     NUMBER
24005  FOR N = 1 TO LN
24010  GOSUB 15000
24020  NEXT N
```



```
28100  FOR H = 1 TO LN
28110 P = 1
28120  FOR B = 1 TO LN
28130  IF AVG(H) > AVG(B) THEN P = P + 1
28140  IF AVG(H) = AVG(B) AND H > B THEN P = P + 1
28150  NEXT B
28160 X(P) = H
28170  NEXT H
28180  IF Q = 1 THEN  PR# 1
28200  GOSUB 11000
28210  PRINT : PRINT "********** RANKED AVERAGES *********"
28220  PRINT : PRINT "CLASS      STUDENT                CURRENT"
28230  PRINT "RANKING       NAME                 AVERAGE"
28240  PRINT : FOR R = LN TO 1 STEP  - 1
28245 K =  ABS (R - LN) + 1
28250  PRINT K;
28260  PRINT  TAB( 10)NAME$(X(R));
28262  IF M(X(R)) < 1 THEN 28270
28263  PRINT " ";
28264  FOR I = 1 TO M(X(R))
28265  PRINT "*";
28266  NEXT I
28270  PRINT  TAB( 36)AVG(X(R))
28280  IF K / 13 =  INT (K / 13) THEN  GOSUB 19000
28290  NEXT R
28295  IF Q = 1 THEN  PR# 0
28300  PRINT : PRINT "DO YOU WANT TO SEE THEM AGAIN";
28310  INPUT C$
28320  IF C$ <  > "Y" AND C$ <  > "YES" THEN 20040
28330  GOTO 28200
29005  FOR N = 1 TO LN
31000 Q = 1
31005  GOTO 20040
31100 Q = 0
31105  GOTO 20040
32000  END
```

Seven Fine Computer Stores!

Save More Than 20%

NORTH STAR — INTERTUBE
THINKER TOYS — HUH

The smartest computers at the smartest price.

Double Density	List	Our Price
HORIZON-1-16K kit,	$1599	$1274
Assembled & tested	$1899	$1519
HORIZON-2-32K kit,	$2249	$1799
Assembled & tested	$2549	$2039
HORIZON-2-32K Quad Density		
Assembled & tested	$2999	$2399
PASCAL for NORTH STAR on Disk		$ 49
Powerful NORTH STAR BASIC		FREE
THINKER TOYS DISCUS/2D	$1149	$ 949
MEASUREMENT SYSTEMS		
ASM Memory 64K		$ 640
Top Quality + Top Speed + Lowest Price		
INTERTUBE II Smart Terminal	$ 995	$ 780
TRS-80 to S-100 HUH Kit	$ 295	$ 260
Assembled & tested	$ 375	$ 350
HUH MINI 8100 Kit	$ 115	$ 103
Assembled & tested	$ 155	$ 139
MARYELLEN Word Processor		$ 38
Best Features of IBM & NORTH STAR		
VERBATIM DISKS		$ 29
Which Computers Are Best?		
BROCHURE........		FREE

AMERICAN SQUARE COMPUTERS
Kivett Drive, Jamestown, NC 27282
(919) 883-1105

CIRCLE 107 ON READER SERVICE CARD

TRS-80 disk software

DATA BASE MANAGER You can maintain a data base & produce reports without any programming. Define file parameters & report formats on-line. Almost use up the required 32K memory. $49.

ACCOUNT manage client accounts & account receivable. Automatic billing & transaction recording. 32K req. $59.

WORD PROCESSOR Specifically designed for the TRS-80. Written in BASIC. No special hardware and text limit. $39.

INVENTORY 9-digit key for fast on-line random access. Reports give order info, performance summary, etc. $39.

MAIL print reports & labels sorted by any field. Random access. special buffering technique. $35.

KEY RANDOM-ACCESS UTIL hashing, blocking, buffering, auto I/O error retry. $19.

CIRCLE 160 ON READER SERVICE CARD

MICRO ARCHITECT
96 Dothan St.
Arlington, MA 02174

FREE! up to $170 in merchandise with the purchase of PET—CBM item!!!

		FREE MERCH.
PET 16K Large Keyboard	$ 995	$130
PET 32K Large Keyboard	$1295	$170
PET 8K	$ 795	$100
PET 2040 Dual Disk *(343K)*	$1295	$170
PET 2023 Printer *(pres feed)*	$ 849	$110
PET 2022 Printer *(trac feed)*	$ 995	$130

KIM-1 $159 (Add $30 for Power Supply) SYM-1	$229.00
2114 L 450 4K Static RAM	6.95
2716 EPROM (5 Volt)	38.00
6550 RAM *(for 8K PET)*	16.20
6502 Microprocessor Chip	9.75
6522 VIA	9.75
6520 PIA	10.50
Auto-Repeat Hardware for PET	24.50
Kite Fight - 2 player PET action game	7.95
Microchess 2.0 for PET or APPLE	17.90
PET Word Processor - Machine Language	24.00

3M "Scotch" 8" disks	SALE	10/$31
3M "Scotch" 5" diskettes		10/$35
Verbatim 5" diskettes		10/$28

Cassettes *(all tapes guaranteed)*
Premium quality, high output lownoise in 5 screw housing with labels:
C-10 10/5.95 50/25.00 100/48.00
C-30 10/7.00 50/30.00 100/57.00

WRITE FOR 6502 AND S-100 PRODUCT LIST

A B Computers 115-B E. Stump Road
Montgomeryville, PA 18936
(215) 699-8386

CIRCLE 101 ON READER SERVICE CARD

Here an **APPLE**
There an **APPLE**
Everywhere an **APPLE! APPLE!**

AND GET A LOAD OF THIS DEAL!

WITH EVERY PURCHASE OF AN **APPLE II-16K** at **$1195.00**

We will sell you an **ADDITIONAL 32K* of Memory** for the incredible low price of **$99.95**

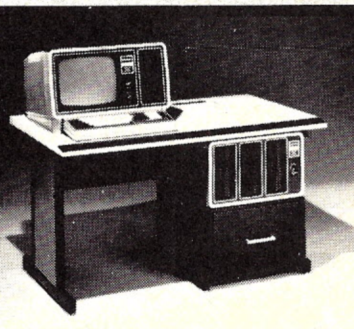
MASTER CHARGE & VISA ACCEPTED

COMPUTER CORNER OF NEW JERSEY
240 Wanaque Avenue, Pompton Lakes, N.J. 07442

CALL TODAY **(201) 835-7080**

*PRIME TESTED MEMORY CHIPS

CIRCLE 123 ON READER SERVICE CARD



TRS-80

SAVE

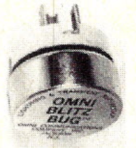
10%, 15% and More on ALL Computers, Peripherals, Software, and ALL other fine Radio Shack® products.

Offered Exclusively By

Radio Shack®
Authorized Sales Center
1117 CONWAY MISSION, TX 78572
(512) 581-2765

NO TAXES on out-of-state shipments.
FREE Surface delivery available in the U.S.
WARRANTIES will be honored by your local Radio Shack® store.

CIRCLE 188 ON READER SERVICE CARD



DOES YOUR COMPUTER SOMETIMES COUGH, SNEEZE OR HAVE A SEIZURE?

It may be suffering from Transiet Glitchitis, a cureable digestive disorder. The Blitz Bug can bring fast relief from these symptoms in less than 50 nano seconds. Available without prescription. Use only as directed.

* No Computer Should Be Without One *

Blitz Bug protects your entire circuit, and plugs into any outlet.
Introductory Price **$14.95**
Reg. **$19.95**
Two for **$25.00**

Omni Communications Co., Inc.
Jackson, New Jersey 08527

CIRCLE 172 ON READER SERVICE CARD

Play a little Sol® Music.

It's not really a piano, of course. But a Sol® small computer system can bring music to your ears as it gives you a strong handle on your business.

Priced from $2500 to $10,000, these are full business systems in every sense — working tools to keep you on top of all that paper work.

Play a little Sol music at our store. Compare Sol to other small computers. We'll show you how much more a Sol system can do for you.

Phone us today. You'll be happy you did.

the computer corner White Plains Mall, 200 Hamilton Ave.
White Plains, N.Y. 10601
(914)WHY-DATA.

CIRCLE 124 ON READER SERVICE CARD

CBASIC Business Software for Microcomputers

Osborne/McGraw-Hill is publishing CBASIC editions of its business software books, **Payroll With Cost Accounting, Accounts Payable and Accounts Receivable** and **General Ledger.** The CBASIC programs in these new editions will run on many floppy disk-based microcomputer systems.

PAYROLL WITH COST ACCOUNTING

ACCOUNTS PAYABLE AND ACCOUNTS RECEIVABLE

GENERAL LEDGER

Each book includes complete source listings with remarks, program and system documentation, and an operator's manual. All this for just $15.00! The softbound Wang BASIC versions of all three books are available now. **CBASIC Payroll With Cost Accounting** is also available now. The remaining books in the CBASIC series will be published in the fall of 1979. The new CBASIC-2 editions are in a convenient, loose-leaf format.

Not a programmer? Don't want to key in thousands of lines of source code from the book? You can buy the programs ready-to-run on floppy disk from one of our 200 dealers. Contact us for the name of your nearest dealer.

CBASIC or Wang BASIC won't work? Here is a partial list of independent consultants who have converted Osborne programs to run on many popular systems using our books as their manuals. Contact us for a complete, up-to-date list of these conversions.

APPLE II:
David McFarling
SMALL BUSINESS COMPUTER SYSTEMS
4140 Greenwood
Lincoln, Nebraska 68504 (402) 467-1878

COMMODORE PET:
Chuck Stuart
CMS SOFTWARE
5115 Menefee Drive
Dallas, Texas 75227 (214) 381-0690

DATAPOINT 1500:
Paul Marchetti
OCCIDENTAL COMPUTER SYSTEMS, INC.
6666 Valjean Avenue
Van Nuys, California 91406 (213) 782-3005

DIGITAL GROUP Z80 under OASIS BASIS:
Charles Finn
2993 Boring Ridge Drive
Decatur, Georgia 30034 (404) 289-8969

MICROPOLIS, MOD II
VECTOR GRAPHIC MZ (REQUIRES CP/M):
David Price
DAP SYSTEMS
3901 Victoria Lane
Midlothian, Virginia

MICROSOFT disk BASIC:
Dan Kindred
GNAT COMPUTERS
7895 Convoy Court
San Diego, California 92111 (714) 560-0433

NORTHSTAR BASIC:
Isaac Bornstein
COMPUTER SERVICES OF ENCINITAS
341 Willowspring Drive
Encinitas, California 92024 (714) 436-2486

TECHNICAL SYSTEMS CONSULTING BASIC, FLEX OPERATING SYSTEM:
Roger Synder
GREAT PLAINS COMPUTER COMPANY, INC.
3460 Taylor View
Idaho Falls, Idaho 83401 (208) 529-3210

TRS-80:
Irwin Taranto
TARANTO & ASSOCIATES
P.O. Box 6073
San Rafael, California 94903 (415) 472-1415

WANG BASIC ON FLOPPY DISK:
Sam Burel
DECOR, INC.
P.O. Box 111
Fairfield, Alabama 35064 (205) 785-8000

BASIC for ZILOG MCZ, ZDS:
Timothy Fossum
CUSTOM COMPUTER SYSTEMS
4328 Pennington Lane
Racine, Wisconsin 53403 (414) 554-0320

CONSULTANTS • COMPUTER STORES • SOFTWARE HOUSES

You can be a dealer for the magnetic surface of Osborne's CBASIC-2 software. We will sell the programs on 8" floppy disks to dealers, consultants and computer stores only. The disk for each book sells for $250.00. Once you buy the floppy disk you may copy it as many times as you like, resell it, change the programs, or use it for your own purposes. But the best part is that we have already written your operator's manuals — our books! We grant you the unrestricted right to use, transfer, or otherwise deal in any way with the machine-readable form of our BASIC programs. We copyright only the printed word in our books.

Order Form

Name: ____________________
Address: ____________________
City: ____________________
State: ____________ Zip: ________
Phone: ____________________

Title	Price	Quantity	Amount
22-5 Payroll With Cost Accounting - CBASIC	$15.00		
23-3 Accounts Payable/Receivable - CBASIC	15.00		
24-1 General Ledger - CBASIC	15.00		
09-8 Payroll With Cost Accounting - Wang BASIC	15.00		
13-6 Accounts Payable/Receivable - Wang BASIC	15.00		
20-9 General Ledger - Wang BASIC	15.00		

Shipping (Shipping for large orders to be arranged)
- □ All foreign orders $4.00 per book for airmail
- □ $0.45 per book 4th class in the U.S.(allow 3-4 weeks)
- □ $0.75 per book UPS in the U.S. (allow 10 days)
- □ $1.50 per book special rush shipment by air in the U.S.

California residents tax	
Shipping	
Total Amount Enclosed	

Calif. residents add 6% sales tax.
S.F. BART residents add 6-1/2 % sales tax.

OSBORNE/McGraw-Hill, Inc.
630 Bancroft Way, Dept. L11
Berkeley, CA 94710 USA
(415) 548-2805 TWX 910-366-7277

□ I am an end user interested in running CBASIC programs on disk. Please notify me of the closest dealer for:

□ I am a dealer interested in purchasing CBASIC programs on disk for resale. Please send description and order information for:

- □ CBASIC Payroll With Cost Accounting
- □ CBASIC Accounts Payable/Receivable
- □ CBASIC General Ledger

Please send the following information:
- □ Becoming an Osborne dealer
- □ School discounts
- □ List of foreign distributors
- □ Other Osborne publications
- □ Please send me the current conversion list

Payment in advance must be enclosed for purchases of up to $70.00. Invoicing U.S. purchases over $70.00 available upon approval of your account. All foreign orders must be prepaid in U.S. dollars drawn on a U. S. bank.

S1034

Programming Contest

Recently, I was fortunate to give the Awards Address at the University of Wisconsin-Parkside's **Computer Fair III**. An ongoing feature of this fair, so capably managed by Donald Piele, is a computer programming contest. At this year's fair, there were four divisions to the contest: A—Grades 10-12, B—Grades 7-9, Junior—Grades 4-6, and Pee Wee—Grades 3 and under.

Presented below are the rules for the programming contest and the problems for Divisions A and B.

— DHA

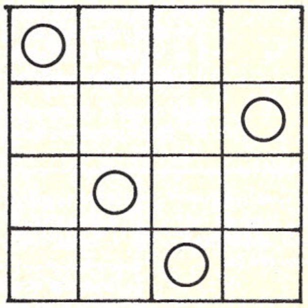
Contest Rules

1. Division A: Grades 10-12. Complete both parts to each problem. Division B; Grades 7-9. Complete part A of each problem.
2. Team Size: You may have up to three members on your team.
3. Computer System: You will be provided a terminal to the HP-2000 BASIC timeshared system; optionally, you may use your own microcomputer system, provided that a hard copy of the programs and sample runs can be made. A printer is available for the Apple II, TRS-80, PET and other systems with RS232 interface capabilities. See us **before** you begin if you need to use such a printer.
4. Contest Problems: Your five team problems are attached to this sheet. You have 2 hours to solve them. An additional 15 minutes will be provided to produce hard-copy output if you bring your own computer.
5. Grading Procedure: Your solutions will be judged on following criteria:
   a. Does it run properly, using the test data requested in the problem?
   b. Is the program easy to read?
   c. Is the program logical, imaginative, creative?

   Ten points will be given for (a), and an additional 5 points will be given for both (b) and (c). No partial credit will be given for a program that does not run.
6. General: No outside help is allowed, including books, programs, or people outside your team. University personnel will be present to answer questions regarding the HP-2000 computer.

1. Patterns

A. Suppose you have a 4 -by- 4 checkerboard and four checkers. You are to place the four checkers on the checkerboard in such a way that each of the four rows, four columns, and two main diagonals contains exactly one checker. One such example is the following pattern:



You are to write a program to print **all** the possible 4 -by- 4 checkerboard patterns which conform to the above rules. Each pattern should be displayed with a star (*) in the position of a checker, and a dot (.) in the position of a space. For example, the above pattern should be displayed as follows:

```
* . . .
. . . *
. * . .
. . * .
```

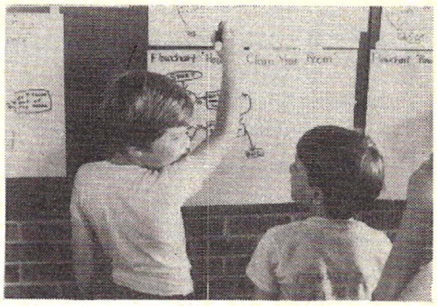
(Hint: To make the display more readable, you should print a blank between each of the characters on a line.)

B. More generally, your program should work for **any** n-by-n checkerboard, with the n checkers placed such that each of the rows, columns and two main diagonals contains exactly one checker. Your program should allow for input of the size n of the checkerboard. Each run should end by printing the total number of successful patterns found. The program should also request from the user whether the patterns themselves should be displayed; if not, **only** the **number** of successful patterns should be printed.

Test your program with n = 3, n = 4 and n = 5 checkers. Include displays of the actual patterns for n = 4, but do not include them for n = 3 or n = 5.

2. Letter Frequencies

A. In cryptography, it is often necessary to count the number of characters in a string. For example,



the string "ABAX1PX ?XX" has the following letter frequency table:

letter	frequency
A	2
B	1
X	4
1	1
P	1
	1
?	1

You are to write a program to input a string of characters, and to output the string's letter frequency table as above. Test your program on the above example, and on the string 'BAA, BAA, BLACK SHEEP!"

B. Note that in displaying the letter frequency table, the letters should be displayed in **decreasing** order of frequency; that is, letters occurring most frequently should appear at the top of the list, and those occurring least frequently should appear at the bottom.

3. Pascal's Warped Triangle.

A. A variation on Pascal's Triangle is

```
row 1: ——————→ 1
row 2: ——————→ 1 1
              1 3 1
             1 5 5 1
           1 7 13 7 1
         1 9 25 25 9 1
```

Each new entry is obtained from the three entries above it by adding them together, as depicted above. The first and last entries in a row are always 1.

You are to write an algorithm which will produce the first eight rows of the above triangle. For simplicity, you should print each row beginning at the left hand margin.

B. You are to write an algorithm which will print the nth row of Pascal's Warped Triangle for any given input value n (where n < 100). Your program should use the least amount of storage possible.

Test your program by printing rows 2, 6, 8 and 13.

4. Sums of Digits.

A. Certain numbers are divisible by the sum of their decimal digits. For example, 18 is divisible by 1 + 8 = 9.

You are to write a program which will find all three-digit numbers which are divisible by the sum of their digits.

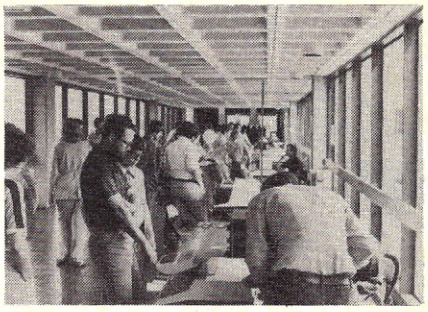
Since there are over 200 of them, your program should merely produce the **number** of successes, rather than a complete list of them.

B. Run your program with n = 2, n = 3, and n = 4.

5. Rock, Scissors, Paper.

A. The children's game of Rock, Scissors, Paper is played by two opponents. At each turn, both players simultaneously signal their choices of either Rock, Scissors, or Paper. The winner of that turn is determined by the rule that Rock wins over Scissors (Rock breaks Scissors), Paper wins over Rock (Paper covers Rock), and Scissors wins over Paper (Scissors cuts Paper).

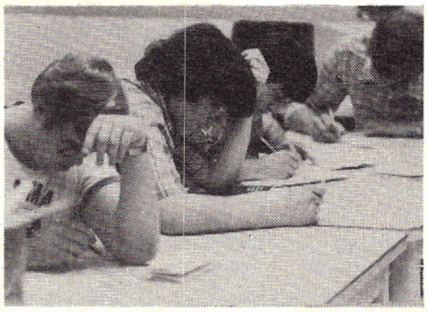
You are to write a program to play the game of Rock, Scissors, Paper between the computer and yourself. At each turn, the computer randomly chooses either Rock, Scissors or Paper (without letting you know). It



then prompts you for your choice. You enter R for Rock, S for Scissors, or P for Paper. The program then prints the winner of that turn. The game continues until you enter a Q to quit when it is your turn. The program should print a summary of the game's results.

Run your program twice, with about 20 turns on each run.

B. In addition, the computer should take into account your playing strategy. It should keep track of your plays, and should make its next choice based on what you have done on your previous turns. For example, if you have chosen P for paper on your last several turns, the computer should be more likely to choose S for scissors on its next turn. The choice of programming this strategy is up to you.



Run your program four times. You should play twice in such a way that the computer will show its ability to use its strategy to win. Then play twice to try to beat the computer, if you can. Use about 30 turns on each run. □

18)9
8
21
3

An Unsyncopated A Cappella Approach to Computing

James W. Cerny and Roscoe Gort

Introduction

There is presently much interest in measuring programmer performance and productivity. The impetus comes from the rising cost of computing personnel compared to the falling cost of computing hardware. We believe a key ingredient in programmer productivity is morale. In fact, all computing people, whether student or teacher, whether amateur or professional, whether programmer, analyst, consultant or manager, are faced with a need to maintain an optimistic outlook despite repeated program failures and schedule disruptions. We have found that a very valuable way to boost morale and build confidence is by singing. This is not a new concept, for we need only think of sea chanties used when bending to the oar. How this can be applied to computing is described below.

Different age cohorts tend to be drawn to different genres of music and we would agree with the maxim, **de gustibus non est disputandum**, that there is no disputing about tastes. For younger programmers, the richest contemporary sources are pop and rock music, constantly broadcast on both AM and FM radio stations. It is easy to draw on the subliminal knowledge of the melodies and lyrics of the popular songs ("Top Forty" hits or "24K Gold" in the patois of the dispensers of this music), locked in the neural patterns of the brain. Middle and senior programmers and managers, who are somewhat older, may be better acquainted with the pop tunes from the 1940's and 1950's. In all cases it is a simple and enjoyable step to provide computing-oriented lyrics and some examples follow.

For ease of reference, most of the songs are discussed in terms of the singer who is best known for popularizing the song, rather than the songwriter. Proper attention must be paid, of course, to the copyright owner of the original music. New lyrics can be copyrighted in their own right, but permission must be sought to use existing music.

Examples

Certain types of popular music are easier to adapt than others. Particularly suitable are songs from the earlier years of rock when tunes were simple and highly repetitive so that a message can be reinforced. Thus, Del Shannon's plaintive "Searching" can be rendered as the universally applicable "Debugging":

> Debugging, I'm always debugging,
> Hoping, some day I'll write,
> A program, a program that works for me...

The Beach Boys' "Shutdown", describing a drag race, can be applied to computer testing situations as "Benchmark":

> It happened on the disks where I/O is fast,
> Two megabyte machines running at full blast...
> Gotta' be cool now, tape mount here I come...

To inspire systems programmers, Jim Croce's "Time In A Bottle", about the elusiveness of time, could be rendered as "JCL in a Proclib" to describe an equally elusive entity. And what computer user could deny feeling a throb of pride when Paul Anka's "Having My Baby" is crooned as "Printing My Core Dump":

> But its printing my core dump,
> What a lovely way of using up printer-time...

For generating general enthusiasm, it is hard to beat the Buddy Holly version of "Oh, Boy":

> All of my programs, all of my coding,
> You'll sure like what you'll be seeing,
> Oh, boy, Oh, boy...

There are also tunes that have stood the test of time somewhat longer. In these days of recycled paper, Bing Crosby's "White Christmas" might inspire "White Paper":

> I'm dreaming of some white paper,
> Just like the stock I used to use,
> Where the perforations tay-er,
> And wood chips are not there...

Or, in rueful contemplation of a vendor price increase, say by IBM, a manager might wish to liken IBM headquarters in Armonk, New York, to San Francisco:

> I left my wallet,
> In Ar-monk, N-Y.
> From a ledger sheet,
> It calls to me...

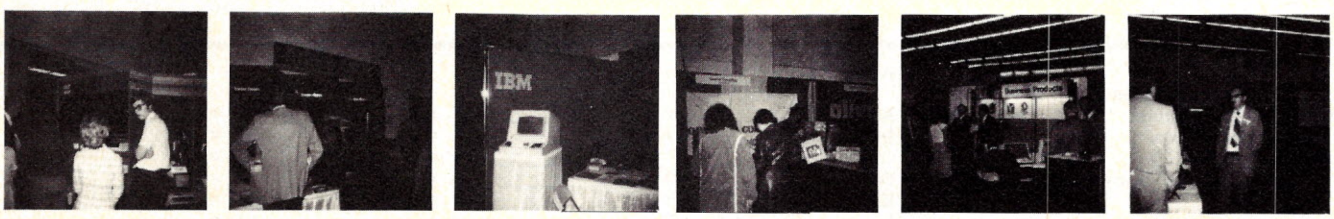
We could go on and on, but much of the interest in this technique is in locally developed "solutions" rather than external "software."

Analysis

At present there is no statistical evidence to support our subjective observations about the effectiveness of such songs. However, we are conducting a factorial experiment with complete randomization to study all the relevant independent factors affecting programmer attitude: sex of listener, age of listener, salary of listener, decibel level of the music, size of the room in which the music is heard, number of times per day the music is played, genre of the music, quality of audio equipment used (including electronic synthesis), sex of singer, weather, time of day, season of the year and state of the national economy. Of course all interactions as well as main effects will be studied for significance. It may take a decade to conduct the experiment, to await the arrival and departure of enough employees to achieve a satisfactory large sample, but we hope to report the results in this same forum when the analysis is complete. □

James Cerny and Roscoe Gort, Office of Academic Computing, University of New Hampshire, Durham, NH 03824.

SOUTHWEST COMPUTER CONFERENCE



Photos by Nancy Wood.

This, the first small computer conference in Oklahoma City, attracted over 7000 visitors. Exhibitors, both large and small were represented including Apple, Radio Shack, IBM, Olivetti, Eastman Kodak and, of course, Creative Computing. E.Z. Million, the affable conference chairman (right photo) is on the faculty of Oklahoma State University Technical Institute.

You Can Be A Winner!

$2000 in Prizes!

Best Ever Computer Trivia Contest

169 Prizes!!

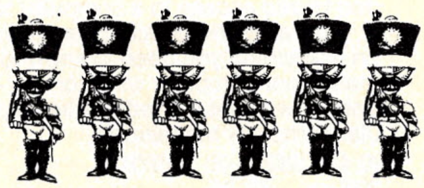
Most readers who have been with us for more than a few months know that we review a great number of consumer electronics products. Most of these are returned to the manufacturer as soon as the review is completed, however, occasionally the manufacturer allows us to keep the product. We seem to have accumulated a substantial number of electronic games (18), hard cover science fiction books (27), computer-related books (42), computer art lithographs by Tom Huston (51) and some other neat stuff. It's taking a lot of valuable space so we decided to give it all away as prizes in a computer trivia (but not trivial) contest.

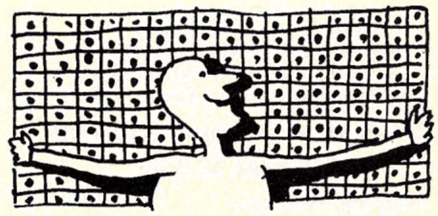
First Prize: Craig M-100 Foreign Language Translator and English/Metric Calculator with 4 extra language modules. Retail value $300. Courtesy of Craig Electronics.

Second Prize: Electronic Mastermind by Invicta.

Third Prizes (17): Electronic or battery action game.

Honorable Mentions (150): Book or art print.





Trivia:

A piece of trivia that might be of interest: we used ten reference books or manuals to determine the values of the variables (actually constants) in the equation. We checked all values in at least two sources which required an additional fourteen books or manuals. — DHA

$$\frac{\Psi - W - \Phi - \left[a \times \sqrt{S + \Sigma \times (\mu + \Theta) \times P \times T}\right] - \pi + \left(\frac{F+M-X}{D-N}\right)}{\sqrt[\Omega-\nu]{G - \epsilon - [\Delta \times (A-Q)] + \sqrt{Y + (C-L)^{\sqrt{Z}} - R}}} \times \frac{\frac{K}{E} \times \omega \times [U - (B - I)]}{\left(\frac{\sigma - \delta + J}{\tau}\right) \times [\xi \times H - (\lambda - \nu)]} \times \frac{\Gamma}{\gamma \times k} = ?$$

Rules:

1. Determine the value of the 45 constants (variables in the equation) to the nearest integer. The intended meanings are straightforward and literal; there are no tricks in the definitions or hidden meanings.

2. Substitute the value for each variable into the equation, perform the arithmetic and calculate the answer. Standard mathematical usage is in effect. Only the positive root of a square root radical should be considered, i.e., 36 should be evaluated as 6. Note: if you evaluate Ω-v as 6, you are required to calculate the 6th positive root of the expression under the radical symbol.

3. Do not call us with questions of interpretation, references or with last-minute entries.

4. Entry blanks must be filled out completely including not only the answer, but also each of the 45 variables. The answer should be expressed as either an integer or proper fraction, as the case may be. The winning entry will be the one that correctly states the answer; if no entry is correct, then the winner will be the entry with the greatest number of correct values for the 45 variables. In either case, ties, if any, will be broken by random drawing.

We have a notion that we will receive few, if any, correct entries. Hence, given that we have 169 prizes worth over $2000, it is probably worth your while to enter even if you do not think you have gotten the correct answer.

5. Entries must be submitted on an official entry blank from Creative Computing magazine (xerox and other copies are not acceptable) and received in our offices (not postmarked) by November 1, 1979. You may enter as many times as you wish. Do not include orders, letters, or anything else in the same envelope. Send to: Creative Trivia Contest, 93 Washington Street, Morristown, NJ 07960.

Clip this entry blank and mail to:
Computer Trivia Contest
93 Washington Street
Morristown, NJ 07960
Entries must be received by November 1, 1979.

Name ______________________

Address ______________________

City ______________________

State ______________ Zip ______

Answer = ____________

A = ______	M = ______	Z = ______	Ψ = ______
B = ______	N = ______	a = ______	Ω = ______
C = ______	P = ______	γ = ______	τ = ______
D = ______	Q = ______	Δ = ______	ν = ______
E = ______	R = ______	ϵ = ______	σ = ______
F = ______	S = ______	θ = ______	ω = ______
G = ______	T = ______	λ = ______	ξ = ______
H = ______	U = ______	μ = ______	δ = ______
I = ______	V = ______	k = ______	Γ = ______
J = ______	W = ______	π = ______	
K = ______	X = ______	Σ = ______	
L = ______	Y = ______	Φ = ______	

You can be a winner!

A Year of birth of Fibonacci
B The Sorbonne is founded (year)
C Number of Shostakovich symphonies
D Year of Napier's 50th birthday
E While in his teens, Gauss settled a 2000-year old question by constructing a polygon with___sides
F Mean radius of the earth in kilometers (Nearest kilometer. Mean radius is the radius of a sphere having the same volume.)
G A college student in 1955 would believe this to be the velocity of light in a vacuum (standard ref. is the Handbook of Chemistry and Physics)
H Atomic weight of fluorine
I Melting point of common salt (°C)
J Number of seconds before tapes self-destruct on Mission Impossible
K Decimal value of 1010101 (binary)
L The Digital Equipment Corp. B171 Flip Chip has___diode inputs
M 549,755,813,888 is 2 raised to what power
N Published absolute maximum collector-to-base voltage (at 25°C) of the GE 2N377 switching transistor.
P To convert from microvolts to gigavolts, multiply by___
Q Number of satellites of Jupiter (observed through 1970)
R Microprocessor used in first Altair
S To convert from hours to nanoseconds, multiply by___
T In the computer game, how many Mugwumps are in hiding
U Cube root of 160,103,007
V Number of letters in Venn's first name
W Number of ways 50 can be written as the sum of three positive integers in nondescending order
X Baud rate of ASR-33 Teletype
Y Number of bits in a 2708 EPROM
Z Price of first class letter in 1962 (cents)
a Sum of the 10 digits of Creative Computing toll-free number
γ Number of rooms an arrow can shoot in Wumpus game
Δ Maximum rated clock speed of Z80A in megahertz
ε Largest value allowed in Apple integer Basic
θ Value at which Fahrenheit and Celcius temperature scales are equal
λ First year of Creative Computing
μ Decimal value of letter A in ASCII code
k Number of cardinal points on a viking compass
π 3 is to 9 and 18 as 2 is to 8 and___
≤ Price of the last item in the group: Linament 12c, Toga 20c, Onion 15c, Car 3c, Pizza
Φ Next term in the series: 7,12,27,72
Ψ The opera Carmen was first performed (year)
Ω Number of letters in leading Orange-growing state in USA
ϒ States admitted to USA before Oregon
υ The first digital computer (Mark I) was completed in (year)
σ Duration of tenth manned space flight (minutes)
ω Fifth lunar landing was Apollo___
ξ Robert Indiana's "Love" lithograph appeared on a US stamp whose first day of issue was January
δ Birth date of the author of the novel in which John Galt is the hero
Γ Value of F in hexadecimal

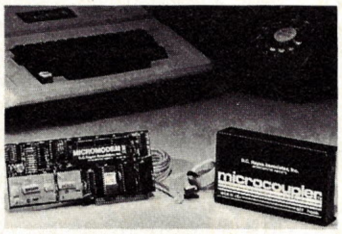
"I have little patience with scientists who take a board of wood, look for its thinnest part, and drill a great number of holes where drilling is easy."

Albert Einstein

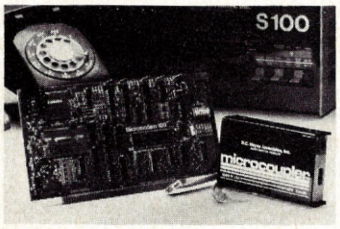
WE'D LIKE TO SHARE A FEW WORDS WITH YOU...

modem / mō' dəm / n: A device for transmission of digital information via an analog channel such as a telephone circuit.

Micromodem II* / mī' krō•mō' dəm tü /*trademark* — a complete data communications system for the Apple II** Personal Computer, combining functions which formerly required a modem, an automatic calling unit, and serial and parallel interfaces. Onboard ROM firmware provides for remote console, terminal mode, and simplified implementation of more sophisticated applications with BASIC programs. The Micromodem II comes with the FCC registered Microcoupler, operates at 110 or 300 baud (Bell 103 compatible), and can automatically dial or answer the phone and transfer data.



Micromodem 100* / mī' krō•mō' dəm wun hun' drəd / *trademark* — a complete data communications system for S-100 microcomputers, providing all the capabilities of a serial interface card and an acoustic coupler, with the addition of programmable automatic dialing and answer. The Micromodem 100 comes with the Microcoupler and is fully S-100 bus compatible including 16-bit machines and 4 MHz processors. The Micromodem 100 operates at either of two software selectable baud rates — 300 baud and a jumper selectable speed from 45 to 300 baud.



acoustic coupler / ə•küs' tik kup' lər / n: A modem that works through the standard telephone handset, transmitting data through the regular earphone and microphone. It can be affected by room noise and suffers from the distortion inherent in the carbon microphone.

Microcoupler* / mī' krō•kup' lər / *trademark* — an FCC registered device that provides direct access to the telephone system without the losses or distortions associated with acoustic couplers and without a telephone company supplied data access arrangement.

D.C. HAYES ASSOCIATES, INC.
MICROCOMPUTER PRODUCTS
10 Perimeter Park Dr. Atlanta, GA 30341 (404) 455-7663

* **Micromodem II, Micromodem 100,** and **Microcoupler** are trademarks of D.C. Hayes Associates, Inc.
** Registered trademark of Apple Computer, Inc.

CIRCLE 137 ON READER SERVICE CARD

IRIDIS™

Programs for your **ATARI®**

- **IRIDIS** is a monthly cassette magazine of programs for the **ATARI 400** and **800** personal computer.
- You get four excellent programs each month on a high quality C-30 cassette, ready to load and run.
- Each issue of **IRIDIS** has an animated "Front Cover" that will delight your eyes!
- With each cassette you get **IRIDIS Notes**, a lively newsletter of information about the **ATARI** computer. You may disagree with our opinionated editors, but you'll never be bored!
- **IRIDIS** brings you fun and games, education, and business programs and "software tools" that will help you enjoy your **ATARI**.

☐ 12 issues for $39.95 in US & Canada
☐ 4 issues for $14.95
☐ Sample copy $4.95

We Accept VISA and Mastercharge

Name ____________

Address ____________

City ____________

State/Zip ____________

IRIDIS, Box 550, Goleta, CA 93017

CIRCLE 151 ON READER SERVICE CARD

TRS - 80 - L1 - 4K
Elementary Education Package #1
$10.00
CROSSWORD (Graphics)
4 Puzzles - 20 Simple Words Total
SPACE SHIP MATH DRILL
I and II (Graphics)
ARRAY MATH DRILL
Count Squares To Find Answer
FIND IT
Prints The Table For Random Problem
Educomp
920 N.W. 179 Terrace
Miami, Florida 33169

CIRCLE 143 ON READER SERVICE CARD

MICROPOLIS

CCA DATA MANAGEMENT SYSTEM HIGH SPEED SORT

This high speed Mach. Lang. Sort allows Creative Computer Application's DMS sort and merge to run in at least half the normal time (depending on the file length and sort key size). This program is designed to be used with the DMS programs. It is very easily interfaced.

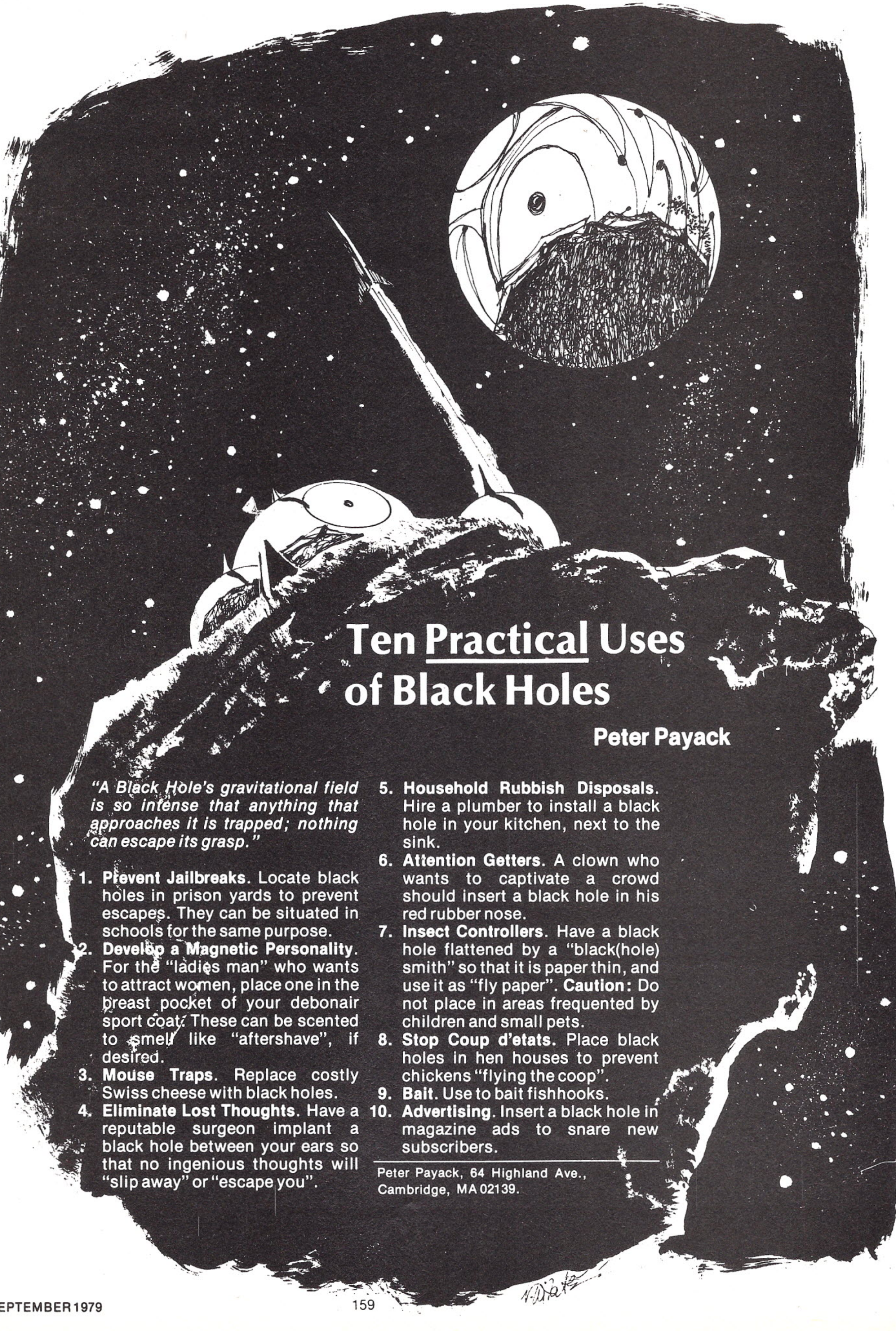
ALSO AVAILABLE: a machine Language Sort Utility. The utility is designed to be easily interfaced to **YOUR** BASIC programs. The sort is 100 times faster than the best written BASIC sort. We also offer a Mach. Lang. Mailing Label Program. To sort the maximum 1000 records requires 15 seconds. The output options allow for sorting by the last name or company name, zip code, and a user defined variable.

COMPLETE USERS MANUAL	**$10.00**
MAILING LABEL (ML)	**79.50**
MAILING LABEL (BL)	**39.50**
MACH. SORT UTILITY	**59.50**
DMS SORT PACKAGE	**69.50**

Available to be shipped immediately upon receipt of your order and payment (money order, cashier's check, Visa or Master Charge). A shipping charge of $2.00 will be added to all orders. California residents, please add 6%.

COMPUTER SERVICES
P.O. BOX 15643
San Diego, CA 92115
(714) 438-9137

CIRCLE 126 ON READER SERVICE CARD



Ten Practical Uses of Black Holes

Peter Payack

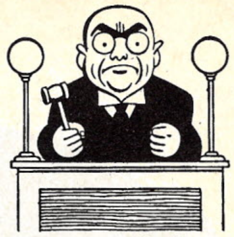
"A Black Hole's gravitational field is so intense that anything that approaches it is trapped; nothing can escape its grasp."

1. **Prevent Jailbreaks.** Locate black holes in prison yards to prevent escapes. They can be situated in schools for the same purpose.
2. **Develop a Magnetic Personality.** For the "ladies man" who wants to attract women, place one in the breast pocket of your debonair sport coat. These can be scented to smell like "aftershave", if desired.
3. **Mouse Traps.** Replace costly Swiss cheese with black holes.
4. **Eliminate Lost Thoughts.** Have a reputable surgeon implant a black hole between your ears so that no ingenious thoughts will "slip away" or "escape you".
5. **Household Rubbish Disposals.** Hire a plumber to install a black hole in your kitchen, next to the sink.
6. **Attention Getters.** A clown who wants to captivate a crowd should insert a black hole in his red rubber nose.
7. **Insect Controllers.** Have a black hole flattened by a "black(hole) smith" so that it is paper thin, and use it as "fly paper". **Caution:** Do not place in areas frequented by children and small pets.
8. **Stop Coup d'etats.** Place black holes in hen houses to prevent chickens "flying the coop".
9. **Bait.** Use to bait fishhooks.
10. **Advertising.** Insert a black hole in magazine ads to snare new subscribers.

Peter Payack, 64 Highland Ave.,
Cambridge, MA 02139.

Software Legal Forum

Harold L. Novick



The old controversy of: "It's my land; no it's not, anybody can use it." has found a new twist. Witness the reader exchange in the Software Legal Forum (June, 1979 Creative Computing) responding to the Open Letter to the CP/M® User's Group by Creative Computing publisher David Ahl (February, 1979). The difference, of course, is that in the new controversy everybody can use and enjoy the property at the same time without first having to exclude others. Welcome, reader, to the World of Intellectual Property.

This column will explore this new land and, hopefully, continue and encourage the dialogue. The interesting aspect is that for any given day you can be simultaneously right, wrong or neither; and then the next day be wrong, right or both, with the whole ball game changed.

For those who came in late, the Ahl—CP/M® User's Group (CPMUG) controversy began when a collection of computer games appearing in a copyrighted book entitled "101 BASIC Computer Games" and in Creative Computing magazine suddenly became available in machine readable form on floppy diskettes. The computer software was reviewed, rewritten in some cases, enhanced and formatted for CP/M® by the CPMUG. According to Dr. Michael Hayes, a participant in the Software Copyright Forum, much of the software distributed was taken from versions that were not usable in the published code, and he believed that Tony Gold and the CPMUG had done the entire personal computing community a big favor by providing usable software. In Mike's opinion, only a software pirate (one who without permission makes a profit from another's software) is a bad guy wearing a black hat. People like Tony Gold, who do it for free, wear white hats, are good guys and are innovators.

Apparently, Creative Computing publisher David Ahl did not think Mr. Gold wears a white hat because he not so subtly threatened a law suit in his Open Letter if CPMUG did not take immediate corrective action. Mr. Ahl said, "Frankly, we're tired of being ripped off and we intend to put a stop to it." Steve North, editor of Creative Computing, obviously agrees. In his replies to the printed letters in the Forum, he opened fire with his phasers and left a whole lot of smoke in his path (but little damage).

Walter Koetke fired right back (and missed!). In his letter in the Forum, Mr. Koetke said flatly that anyone who spends considerable time debugging programs that don't run and adding features to make programs useful in a specialized environment produces a different program that can be distributed without a copyright infringement of the original program. Wrong! As will be made clear in later columns, if the copyright laws apply to what Messrs. Gold and Koetke are doing (and that is a big IF), then that is just as much an infringement as it would be to produce an English version of Remarque's "Im Westen Nach Stille" (All Quiet on the Western Front). Clearly, all is not quiet.

But even in all the noise a quiet whimpering can still be heard. Mr. Barry Walryms stated in the lead Forum letter regarding Creative Computing's charge of infringement, "your position, however legally well based it may be, is ill-advised from both your position and mine." Mr. Walryms apparently believes that just because CPMUG does it almost at cost, no one is hurt. Perhaps he is forgetting that by diminishing the size of the market, the economic return of the legal provider of the software is lessened. This is a recoverable damage and one that has been anticipated in the new copyright act. I'm sorry Suzanne (Suzanne Rodriguez, Editor of Dr. Dobbs' Journal), you are wrong when you stated in your letter in the Forum that just because you don't do it for money it's okay. Furthermore, reducing the incentive of economic return can result in less personal computing software when people refuse to risk their investment and time to produce and document good software.

Steve North's phasers made a slightly damaging hit when he stated in one of his responses that it is their software and they should have the right to give, sell or retain it as they wish. Just imagine the software to be your corn field being invaded by bike riders. Shouldn't you have the right to either let them use the field or to refuse? Answer: If it is your cornfield and you have been given that legal right by your government, then yes you can. Querry: Is software a corn field? Answer: Yes, no, maybe and sometimes. No, this author is not confused, only the situation is. But keep your monitors tuned and a non-confusing answer may reveal itself.

The astitute reader should have noticed by now that this author has not stated who is riding in the Enterprise with a white hat and who is a Klingon with a black hat. If only the game could be that clear. Rather than hatting the participants, the better approach is to welcome each individual perspective into the corn field and then announce to the surprised group that each is legally correct and incorrect.

The principal difficulty in deciding what is correct or incorrect and even whether software can be analogized to a corn field is that software is a brand new type of intellectual property. As such, it's metes and bounds have yet to be defined and even its very legal definition has yet to be decided upon. Another aspect of the problem is that software has many faces. It can be in the form of computer games, computer languages and editors, applied software (which makes the computer an integral link in a system such as a sewing machine), word processing software, data base management software and, perhaps, even the data base itself.

The law of trespass is old and clear and its application to corn fields is fairly well established. Also old and clear are the laws of patents, copyrights, trademarks, trade secrets and contracts. But their application to software is not well established.

The foregoing topics will be addressed again in forthcoming issues of Creative Computing magazine. The reader's interaction and participation will be continued and those wishing to engage in the dialogue should send their letters to this magazine. As for this author, his phaser banks have been depleted and he must return to his home base to recharge them before more can be done.

The comments and opinions of the author are given for educational purposes only and are not meant to be legal advice. Specific legal question's should be referred to your personal attorney. □

Harold L. Novick, Patent Attorney, Larson, Taylor and Hinds, Arlington, Virginia 22202.

Q & A
The "Tiny" Interpreter Exercise

Philip Tubb

Over the years there have been numerous requests for an article describing the internal workings of interpreters. The following will provide an interesting self-teaching exercise for individuals wanting to explore interpreters and should also be an ideal classroom exercise. The experimentation and learning shouldn't stop after entering and running the program in the article. Go ahead and add new features and commands to the "Tiny" Interpreter... and really start learning.

A new column, "Operating Systems Q&A" by John Craig, began in the November-December 1978 issue. Many questions were sent in which weren't limited to operating systems, and ultimately I was asked to do a general questions and answers column.

One question particularly struck me as interesting. John Thompson of Monee, Illinois asked about "...that never-never land between machine language and a high level language like BASIC..." and wanted to know how interpreters are written. Most users of personal computers use high level languages, but few know exactly how they work or how they are written. Since most users are familiar with BASIC, I decid d to answer the question by writing an interpreter for a small BASIC-like language. Rather than writing the interpreter in machine language as is usually done, I've decided to write it in BASIC since most readers will then be able to follow the programming logic.

The writing of interpreters is a complex subject, but is of sufficient interest to warrant a detailed explanation. So, this first Q&A column will be a bonus double-size single-topic issue. Henceforth I'll try to answer several questions in each column (unless I find another worthy topic). There is no extra charge for this special issue, so enjoy!

The "Tiny Interpreter" Specs

First, it is important to decide what the language will be like. I'll pick a few BASIC statements, just enough to illustrate some key interpreter concepts: LET, INPUT, PRINT, GOTO and IF. An interpreter needs a command set such as LIST, RUN and RENUMBER. Most BASICs include a very complex floating-point arithmetic package, which is too elaborate for this example (and is really a different subject entirely). I'll stick to add, subtract, less than and equal to with integers from -9999 to 9999. In traditional interactive-interpreter style, I'll write a line oriented editor which allows lines to be added, deleted and replaced by line number.

The next step is to define the exact syntax of the language. I prefer to use a modified form of BNF (an acronym for Backus Naur Form - a notation for describing the syntax of languages). First, I define the element <program>. The definition is approximately:

<program>: :<line>[<program>]

The ": :" is read "is defined as" (this is a slight variation from the standard symbol, ": : =", because it is easier to type). Anything in brackets is optional. In English the definition above is: "a program is defined as a line optionally followed by a program or, in other words, one or more lines" which is why BNF is used instead.

Next, we dive in and define the elements used in the definition of <program>. In this case the only one is <line>:

<line>: :<line number><statement><return

It is not necessary to define the ASCII character return , since this is for our own use. Let's continue defining elements until there are none left to define:

<line number>: : an integer from 1 to 9999

Although an integer from 1 to 9999 can be defined in BNF, it is much easier to read in English. We can use a slash (/) to indicate "or":

<statement>: :<LET>/<INPUT>/<PRINT>/<GOTO>/<IF>

<LET>: :<variable>" = "<value>

<INPUT>: :"INPUT"<variable list>

<PRINT>: :"PRINT"<value list>

GOTO : :"GOTO" line number

IF : :"IF" value "THEN" statement

<variable>: :<A-Z>[<characters>]

Note 1

<value>: :(<variable>/<number>)[<binary op><value>]

<number>: : an integer from -9999 to 9999

<binary op>: :"+"/"-"/"<"/"="

<variable list>: :<variable>[";" <variable list>]

<value list>: :<value>[";" value list>]

That's all, but most languages run on for pages. Anything in quotes is typed literally. <A-Z> is not defined because it's fairly obvious that it's a letter from A to Z. "Note 1" for the <variable> definition would explain that <characters> is one or more ASCII characters not including "+", "-", "<", "=", or ";", that there is some maximum variable name length, and that the name may not contain the word "THEN". One might also wish to mention that for convenience variable names beginning with "INPUT", "PRINT", "GOTO", or "IF" should not be chosen since they cannot be assigned values with a LET statement. (The reasons behind this will be explained as we go along). The use of parentheses in the definition of <value> is to clarify that the definition is "a variable or a number, optionally followed by a binary operator and a value," rather than "a variable, or a number optionally followed by a binary operator and a value."

1	LET
2	INPUT
3	PRINT
4	GOTO
5	IF

Table 1.

Next, we grind the syntax down into something easy to work with. There are five statement types, and they are shown in Table 1. All statements start with a keyword (such as "PRINT"; except LET. All are composed of combinations of 8 key syntax elements, shown in Table 2. This allows us to make a simple syntax list with the keyword, the statement type number, and its key syntax elements (see Table 3). This will be useful when actually writing the interpreter. Although this language is so small it

Phil Tubb is President of ALF Products, Inc. (1448 Estes, Denver, CO 80215), manufacturers of music synthesizer boards for the Apple computer.

may not be necessary to approach syntax formats in a "key syntax elements" scheme, in conventional (larger) languages it is often the easiest way to go.

Usually one must determine the function of each statement in detail, but since this language is essentially BASIC we'll assume the intended function is already known.

How To Write The Interpreter

There are countless ways to write interpreters. There are many factors which must be balanced. Everyone likes fast interpreters, but fast interpreters tend to be large or to be for limited languages. For example, a machine language could probably run faster than a high level language, but who would want an interpreter for a machine language? Everyone likes to see syntax errors printed right when they type in the line, but some languages cannot be syntaxed on entry. Should the lines be stored in text form, or changed to a more compact form? These questions, and many others, must be worked out based on the intended application of the language. The scheme which I prefer is not the most efficient, although it is very efficient; it is not the fastest, although it is very fast; it does not have the most compact storage format, although it is very compact; and it is not easy to write, although there are harder schemes to write. Probably the biggest disadvantage to this scheme is that it makes the interpreter itself harder to write than many other schemes. One should bear in mind, though, that the interpreter need only be written once (or a few times) and must be used countless times.

Everyone likes fast interpreters, but fast interpreters tend to be large or to be for limited languages.

I prefer to analyze the syntax of each line as it is entered and store it in a special, easy-to-use format. All line numbers will be stored as integers from -9999 to -1 (the opposite of the number actually typed). Each line will be a line number, then a number from 1 to 5 indicating the statement type, then one or more numbers that will differ from statement to statement, and finally a -10000 to mark the end of the line. "THEN" will be stored as -10001, as will semicolons in value lists. Binary operators will be stored with the numbers shown in Table 4. Nothing else will be stored except variables. Variables are interesting things to work with. Obviously one would wish to have multi-character variable names (like "SUM"). Another common desire is for names followed by an integer (like "SUM1", "SUM2" and "SUM3"). Apple owners are familiar with the problems associated with multi-character variable names. In Apple's Integer BASIC, the whole variable name is stored as text, requiring a great deal of storage when long names are used frequently. Further, each time a variable is referenced, Integer BASIC scans a long variable value table for a matching name. This takes a long time, so the shorter the name is, the better. Applesoft (Apple's Microsoft BASIC) works somewhat differently. It also stores the whole variable name as text, requiring a great deal of storage. Table search time is reduced by limiting the search to the first two letters of the name. Unfortunately, this makes SUM1 and SUM2 the same variable, since they both start with SU. Rather primitive, I'd say. I like to build a table of variable names. When a line is entered which contains a variable, the interpreter scans through the table. If the name is not already in the table, it is added to the end. To store the variable reference, one need only store an integer which indicates the variable's place in the table. Room can be reserved in the table for the variable's value, too. This scanning takes place only when a line is entered, not while the program is running, so execution speed is kept high (in fact, the integer which indicates the variable's location in the table is quite useful for determining where the value should be stored or read.) With this scheme you get fast execution, and the space required for any variable typed in a line is the same, regardless of the number of letters in the name (except that the name must be stored once in the table).

The storage format just described is useful for (a) reducing the amount of memory required to store a line, and (b) increasing program execution speed since the interpreter need only look at numbers like "3" rather than strings of characters like "PRINT". However, there remains one particularly tedious operation which must still be done during program execution. "GOTO 50" would require that line 50 be located. Fortunately, there is a simple solution to this problem. At the beginning of RUN, we just zip through the program finding all the lines, and replace the stored numbers with an integer that indicates the location of the line. These "pointers" will be stored as non-negative numbers, so as to not be confused with actual numbers, which are stored as negative numbers. This scheme has at least two advantages: (a) it speeds up program execution, and (b) it allows one to be notified of all non-existent line references (e.g., GOTO 50 when there is no line 50) at once, without waiting for the particular line with the error to be executed.

The main disadvantage of all this string/number crunching is that it takes time. Some people claim that it takes additional programming in the interpreter but I'm not sure just how much extra, if any, is required. Fortunately, it's not too bad to take time at this particular point because (a) each line need only be crunched once, rather than each time it is run,

1	(variable)
2	(value)
3	(variable list)
4	(value list)
5	"="
6	(line number)
7	"THEN"
8	(statement)

Table 2.

and (b) the user may not even notice the slight pause when he presses return because he is preparing to type the next line, and if he is using a printer the crunching can almost always be done while the printer is doing the line feed.

Microsoft: The Good and the Bad

Now we're ready to begin writing. In real life I wrote the interpreter once, threw it away, and began again with a good idea of what would be coming up (this is my standard practice). Since I happen to have an Apple II computer handy, the interpreter is written in Applesoft. I would like to digress momentarily to discuss some difficulties I had using Applesoft. I started programming several years ago with Hewlett-Packard 2000 Timeshare BASIC, which is an excellent BASIC. HP's strings are far easier to use than Microsoft's LEFT$, RIGHT$ and MID$ (plus, HP's method is preferred in recent proposed ANSI standards). For example, to delete the first character in a string, you use: 100 A$ = A$(2). With Microsoft, you must use: 100 IF LEN(A$) = 1 THEN A$ = " " : GOTO 120 and 110 A$ = RIGHT$(A$, LEN(A$)-1). I wouldn't even begin to translate HP's 100 A$(4,7) = B$(20,23), especially if the length of B$ might be 21 characters. These string operations got to be so tedious, that in the rewrite I decided to give up and avoid using Applesoft's strings at all (which will probably give

HUNGRY?!

Feel The Need For Some Delicious, Nutritious, Organic Software For Your Apple Computer?

THE HARVEST IS IN!!

The Apple Orchard Has Taken Root, Blossomed, And Is Now Harvesting A Choice Crop Of Softwares.

*** VOLUME I IS NOW AVAILABLE ***
(At least 17 programs in each volume.)

GAMES
SOUND
GRAPHICS
UTILITIES
EDUCATION
OPERATION HINTS

Fully documented with YOU in mind.

APPLE COOKBOOK How to get the most from your computer.
APPLESAUCE Tasty tidbits of software fully documented for your pleasure.
SPICED CIDER Information on future releases of software.

Magazine also comes in cassette. (Please specify if you want cassette.) Price: $15.00.

APPLE ORCHARD
131 HIGHLAND, VACAVILLE, CALIF. 95688
707-448-9055

CIRCLE 109 ON READER SERVICE CARD

SPECIAL OFFER

$2.99 **$2.99**

The HANDCALCULATOR HANDBOOK
by T.R. DICKSON

364 Pages **25 Chapters** **Paperback**

This Handbook is a complete calculator reference book. A reference containing a wealth of algorithms for applications of typical scientific calculators. Nearly 500 program examples illustrate a wide variety of calculator uses. Programs are given in RPN (Reverse Polish Notation), AOS (Algebraic Operating System) and Algebraic number processing illustrating the use of any typical calculator. Each algorithm is explained and described in a flow chart type of notation. This notation allows the direct and specific implementation of the algorithm on a calculator. Furthermore this flow chart notation can be used as a direct aid in the computer implementations of the algorithms.

Table of Contents by Chapters

Table of Programs • How the Calculator Works • History of the Handcalculator Glossary of Calculator Terms • Calculator Functions and Operators • Types of Calculators • Computations on the Calculator • Conversion Constants • Programming for a Calculator • Function Evaluation • Tabulations • Logarithms • Powers and Roots of Numbers • Roots of Functions • Vector and Matrix Operations • Solving Linear Simultaneous Equations • Statistics • Curve Fitting • Numerical Integration Trignometric Functions and Triangles • Conversions • Complex Arithmetic • Surveying Navigation • Business and Finance • Miscellaneous Applications • References

Usual list price is $5.95 but the Handbook can be yours for only $2.99 by direct mail. Price includes handling and mailing charges. Add 70¢ per book if UPS shipping is desired. Order this valuable reference NOW. Money back guarantee if not satisfied.

ALCHEMY PRESS
512 Monterey Drive Aptos California 95003

Enclosed please find a check or money order in the amount of $ ________ for ___ copies of the HANDCALCULATOR HANDBOOK. (CA residents add 6% sales tax)

NAME ____________________

ADDRESS ____________________

CITY/STATE/ZIP ____________________

CIRCLE 106 ON READER SERVICE CARD

MICROCOSM I

30 ORIGINAL TRS-80 PROGRAMS
ON HIGH QUALITY CASSETTES
WITH USERS MANUAL AND STORAGE CASE
FOR LESS THAN 67¢ EACH
YOU CAN HAVE ALL 30 16K LEVEL II PROGRAMS

GAMES - News Photographer, South Pole Driver, Atlantis and 5 more
EDUCATION - Music Transposition, Spelling Drill Country Identification and 3 more
TRS-80 DIAGNOSTICS - Instruction Tests Memory Tests
HOME - Flowering Houseplants, Mortgage Calorie Counter and 4 more
JUST FOR FUN - Magic Squares, Factorial Biorhythms and 3 more

THE COMPLETE MICROCOSM I PACKAGE
IS AVAILABLE FOR ONLY $19.95
FROM YOUR LOCAL COMPUTER STORE
OR DIRECTLY FROM
BASICS AND BEYOND INC
BOX 10
AMAWALK, N. Y. 10501
914-962-2355
THERE IS NO CHARGE FOR POSTAGE AND HANDLING
N.Y. RESIDENTS ADD 5% SALES TAX

CIRCLE 112 ON READER SERVICE CARD

SUPER SPECIAL
Apple II 16K
$999.99

(8) 16K Rams	$65.00
(10) Verbatim Disks	$27.00
Axiom 820 Microplotter	$699.99
Axiom 800 Printer	$385.00
TTY Model 43 W/RS232 Interface	$1099.00
Integral Data Products	
IP-125	$739.00
IP-225 with all options	$1099.00

The Computer Stop
16919 Hawthorne Blvd.
Lawndale, CA 90260
(213) 371-4010
Tues.—Sat.
11:30 to 6 PM

CIRCLE 127 ON READER SERVICE CARD

you a better idea of how it's done in machine language anyway). Another difficulty I had was that if I used 100 FOR A = 1 TO B in HP BASIC, and B is less than 1, the body of the loop (between the FOR and the NEXT) would not be done at all, but with Applesoft it will be done once anyway. Generally I solve this by adding an IF statement which should be unnecessary (according to the infallible Phil Tubb). Just to give you an idea how useful a zero-times-through loop can be, we'll put three asterisks (***) in the text each place it would have been useful. I found it necessary to move all my subroutines to the beginning of the program because Applesoft doesn't use the pre-scan GOTO technique discussed previously. Normally, I like to place my subroutines at the end of the program (who doesn't?). There were some instances where variable names mixed with keywords; I'll point those out later for those of you who enjoy seeing others make interesting mistakes. Finally, I had originally planned to separate variable names or values with commas (like 10 INPUT SUM1, SUM2, SUM3) but Applesoft doesn't allow strings with commas to be input. "Surprise" doesn't even begin to describe my emotions. Programming of this sort deserves nothing but contempt; if I had written it I would hide in shame.

Initial Housekeeping

Onward with the actual interpreter example program. First we need to set some limits on variable size and similar parameters. By gathering all such parameters together to the beginning of the program, the parameters can be easily changed later if needed. We need to know the maximum program size (MPROG), the maximum line length (MLINE), the maximum variable table size (MTABLE), the maximum variable name length in characters (MVAR) and the maximum input line length in characters (MIN). We also need to dimension arrays and set a few initial variable values. A pointer is needed to the next available space in PROG%, the program array (NPROG); a pointer to the next available space in TABLE%, the variable table (NVAR); a flag for run mode/syntax mode (RFLAG); and a temporary NVAR variable (QNVAR). The BASIC programming is as follows:

```
10 MPROG = 500 : MLINE = 40 :
   MTABLE = 500 : MVAR = 40 :
   MIN = 80
20 DIM IN(MIN + 3),T1(MLINE-1),
   T2(MVAR-1),PROG%(MPROG),
   TABLE% (MTABLE-1) :
   NPROG = 0 : NVAR = 0 :
   RFLAG = 0 : QNVAR = 0 : GOTO
   5000
```

Arrays T1 and T2 are used as temporaries. The PROG% and TABLE% arrays would be all available unused memory if programming in machine language. (For those not familiar with Microsoft BASIC, the % indicates an integer array which requires less storage than a regular floating-point array.)

The first step is to obtain an input line from the user. A subroutine will be used to input a string and convert it into an array of numbers. In machine language, the string would normally appear as a list of numbers. The programming is quite simple:

Applesoft doesn't allow strings with commas to be input. "Surprise" doesn't even begin to describe my emotions.

```
110 INPUT " ",A$ : IF MIN<
    LEN(A$) THEN PRINT "INPUT
    LINE TOO LONG." : GOTO 110
114 IN(0) = 0 : I = 0 : IF LEN(A$)
THEN FOR A = 1 TO LEN(A$) : IN(A-1)
    = ASC(MID$(A$,A,1)) : NEXT A :
    IN(A-1) = 0
116 RETURN
```

I originally typed IF LEN(A$)>MIN THEN... since that is far more clear, but it turned into IF LEN(A$)>M INT HEN. Also *** in 114. A zero is placed as the last item in the array as an end marker (in machine language this might just be the ASCII return character instead). The variable I is set to the first character, and will be advanced as the processing for each character is finished.

The first thing to do with the input line is to check to see if the first character is a digit. If it is, a line is being input; otherwise, it is a command.

```
5000 NVAR = QNVAR : PRINT
     "-"; : GOSUB 110 : IF IN(I)<48
     OR IN(I)>57 THEN 5200
```

Let's assume for the moment that the input line begins with a digit and, therefore, line 5000 doesn't branch to 5200. The first thing to do with a line number is to translate the ASCII characters into a number (like 1, 2 and 3 into 123). Since numbers are used in various places, a subroutine is appropriate. This one will accept numbers from -9999 to 9999.

```
80 N = 0 : N1 = 1 : IF IN(I) = 45
   THEN N1 = -1 : I = I + 1
90 N2 = IN(I)-48 : IF N2<0 OR
   N2>9 OR N*10 + N2 9999 THEN
   N = N*N1 : GOTO 30
100 N = N*10 + N2 : I = I + 1 :
    GOTO 90
```

The ASCII value for "-" is 45, and the code for "0" is 48. Exit from the routine is to line 30, which is a subroutine to delete spaces (ASCII value 32):

```
30 IF IN(I) = 32 THEN I = I + 1 :
   GOTO 30
40 RETURN
```

The reason for this is that something must be done about spaces. Most BASICs ignore all spaces not in quotes or REM statements, which leads them into such follies as turning MIN THEN into MINTHEN and then into M INT HEN. I prefer to allow (and ignore) spaces only between syntax elements. Since a number is essentially a syntax element, the subroutine at line 30 is used to consume any spaces following the number.

Line Numbers

You will recall that only positive numbers are allowed for line numbers, and that they are stored using negative numbers. The subroutine at line 80 returns N as the integer, which must be checked for being positive and stored. The line currently being entered will be stored in the T1 temporary array, and PNT will point to the next available element. Since line numbers are also used by GOTO, a subroutine is required to translate and store a line number:

```
50 GOSUB 80 : N = N : IF N>= 0
   THEN POP : PRINT "ILLEGAL
   LINE NUMBER." : GOTO 5000
60 IF PNT = MLINE THEN POP :
   PRINT "LINE TOO LONG." :
   GOTO 5000
70 T1 (PNT) = N : PNT = PNT + 1
   : RETURN
```

Line 60 is a handy subroutine that stores N into the T1 array, first checking to be sure the line isn't too long. Note that POP causes Applesoft to forget that we're in a subroutine, and allows a return directly to line 5000 (where a new line will be input).

How does this store-a-line-number routine get called? First, we have to check to see if any of the line numbers have been changed to pointers. If so, they must be changed back to line numbers because if a new line is added the locations of existing lines may change. RFLAG will be set

ANNOUNCING

TRS-80* PEOPLE'S PASCAL PROGRAM DEVELOPMENT SYSTEM

"Tiny" Pascal, runs on any 16K Level II system, includes the **programming structuring** capabilities of full Pascal, but not **data structuring.**

Compiled People's Pascal programs run about five-times faster than Level II Basic—graphics run eight-times faster.

People's Software Tape 3 .... **$15.00**
Tape 1 Level II, 34 business, educational programs **$7.50**
(Level I version, 24 programs—separate tape)
Tape 2 Level II, 77 programs from Osborne book, "Some Common Basic Programs." **$7.50**
Tape 4 Level I, business and educational **$7.50**
Tape 5 Level II, business and educational **$7.50**

Add 50c each tape for postage and handling.
California residents add 6% tax. Dealer inquiries invited.

COMPUTER INFORMATION EXCHANGE
Box 158
San Luis Rey, CA 92068

CIRCLE 125 ON READER SERVICE CARD

CASSETTE USERS—*PEEK!*

for surefire, consistent loading with TRS APPLE OTHERS

"PEEK 80" lets you

- **set and monitor best loading level**
- **erect precise azimuth**

$49.50 includes op-amp peak reading meter with shelf life battery, azimuth calibration tape, adjustment tool, patch cord, user's manual.

Order the COOK PK-80 today! And, send for information on the new, ultra reliable Microfusion* Digital Cassettes (the high packing density of a new DuPont chrome digital tape)—cassettes that really work!

*TM Cook Laboratories, Inc.

Cook Laboratories, Inc. 375 Ely Ave., Norwalk, CT 06854
Please send .... PK-80 sets @ $49.50 (total $..................) to
Name
&
Address .... Zip ....
☐ Check or M.O. enclosed (Conn. residents add 7% tax)
Charge to my MasterCharge card ☐ Charge to my VISA card ☐
Card # .... Exp. date ....
Signature ....
☐ Send information on Microfusion Digital Cassettes.

CIRCLE 133 ON READER SERVICE CARD

Verbatim™

FLOPPY DISKETTES

5¼" ONLY $2.50 ea. in 100's

10 FOR $3.00 ea. / 50 FOR $2.75 ea.

8" ONLY $3.25 ea. in 100's

10 FOR $3.75 ea. / 50 FOR $3.50 ea.

SEND FOR OUR FREE ELECTRONIC & COMPUTER PRODUCTS CATALOG

SPECIFY DISKETTE TYPE & COMPUTER
5¼" Soft Sector, 10 sector, 16 sector
8" IBM Compatible, Hard Sector

CALL 24 HRS. TOLL FREE TO ORDER
NATIONAL 800-824-7888
CALIF. ONLY 800-852-7777
ALASKA & HAWAII 800-824-7919
ASK FOR OPERATOR 813

VISA, MASTERCHARGE OR C.O.D.
SEND CHECK, SAVE C.O.D. FEES

D•C SOFTWARE & COMPUTER PRODUCTS
A subsidiary of DataComm Interface Systems
POST OFFICE BOX 503
SAN BRUNO, CA 94066
(415) 348-2387

CIRCLE 138 ON READER SERVICE CARD

HOW TO BUY A BUSINESS COMPUTER

WHEN TO BUY • WHEN NOT TO BUY • HOW TO CHOOSE CONSULTANTS • DATA GATHERING • PACKAGED VS CUSTOM SOFTWARE • CONTRACTS • FINANCING • WARRANTEES AND MAINTENANCE • INSTALLATION MANAGEMENT AND MUCH MORE .....

"SUPER BOOK!" DP Consultant, Mercer Island, Washington.

"I wish I'd had this book before I bought my computer!" TV Station General Manager, California.

"It's the best book I've seen on the subject. Send me 5 copies to send to my clients." CPA, Orange County, California.

"I really learned something from this book and I'm in the business. I'm recommending it to my clients." DP Consultant, Alhambra, California.

Here's a straight forward approach to business computer purchase and installation. It will save you time and money.

ORDER NOW! If not completely satisfied, return within 30 days for a full and immediate refund.

8½ x 11 Softbound ▪ 178 pp. ▪ $12.95
Credit Card Orders 1-805-964-7448

DDC PUBLICATIONS
5386 Hollister Ave., Santa Barbara, CA 93111
Rush____ copies of "Winning The Computer Game" at $12.95 per copy. (CA residents add 6% sales tax)
NAME ________________
ADDRESS ________________
CITY/STATE/ZIP ________________
☐ Check ☐ MasterCharge ☐ Visa
Card No. __________ Exp. Date ______

CIRCLE 139 ON READER SERVICE CARD

to non-zero when pointers are present, and zero when they aren't, so we use:

```
5010 IF RFLAG THEN GOSUB 340
```

This will call a subroutine at line 340 (which coverts pointers back to line numbers) if RFLAG is non-zero. Now for the line number:

```
5020 PNT=0 : GOSUB 50 : IF NOT IN(I) THEN PNT=0 : GOTO 5090
```

If the end marker is found after the line number then a statement is to be deleted and the line shown above branches to line 5090.

At this point a new or replacement statement is being checked. It must be syntaxed (checked for proper format and identified as to type) and stored. Since all statements begin with a keyword except LET, the first step is to check to see if the line begins with any of the known keywords. If it doesn't we'll assume it is a LET statement. The syntax of the various statements, shown in Table 3, is represented in DATA statements as follows:

```
9000 DATA 5,73,78,80,85,84,2,3,255
9010 DATA 5,80,82,73,78,84,3,4,255
9020 DATA 4,71,79,84,79,4,6,255
9030 DATA 5,71,79,32,84,79,4,6,255
9040 DATA 2,73,70,5,2,7,8,0
9050 DATA 1,1,5,2,255
```

```
(LET)1;1,5,2
INPUT3;4

(LET)1;1,5,2
INPUT2;3
PRINT3;4
GOTO4;6
IF5;2,7,8
```

Table 3.

The first number in each line is the number of letters in the keyword. This is followed by that many numbers which indicate the ASCII characters of the keyword. Following these numbers are the numbers given in Table 3. Note that line 9030 is the same as 9020 except it contains an extra letter (ASCII 32) in the keyword. This allows GOTO to be entered as either GOTO (line 9020) or GO TO (line 9030). Each syntax list ends with either 255 or 0, and following the 255 or 0 the next statement type begins. The 0 indicates that the next statement is LET which has no keyword.

First the data must be restored in case this is not the first time a statement is being deciphered. Then the length of the keyword is read and checked to see if it matches the keyword used in the input line:

```
5030 RESTORE
5040 READ L : FOR A=0 TO L-1 : READ B : IF IN(I+A)=B THEN NEXT A : GOTO 5070
```

Line 5040 will branch to line 5070 if there is a keyword match. Otherwise, the read continues until a 0 or a 255 is found. If a 255 is found we go back to 5040 to try the next statement type.

```
5050 READ L : IF L=255 THEN 5040
5060 IF L THEN 5050
```

Now, at line 5070, the statement type is known. The next number in the DATA statements will indicate the statement number, then its syntax. "L" is the number of characters in the keyword, or 0 if no keyword was found (and thus a LET statement is assumed). The data for LET (in line 9050) contains no keyword information and therefore begins with the statement number (1). The next step is to skip over the keyword (if any) in the input line and also skip over any spaces found. Then the statement number must be read from the DATA and stored:

```
5070 I=I+L : GOSUB 30 : READ N : GOSUB 60
```

Next the syntax can be read and an ON GOSUB used to branch to the appropriate routines. This will be done repeatedly until a 255 is found in the syntax list.

```
5080 READ N : IF N<> 255 THEN ON N GOSUB 120,220,260,280, 300,50,320,330 : GOTO 5080
```

When a 255 is found, it falls through to the next line which stores the -10000 end-of-line marker. If the input line doesn't end, then an error is printed.

```
5085 N=-10000 : GOSUB 60 : IF IN(I) THEN PRINT "END OF LINE EXPECTED." : GOTO 5000
```

Syntax Elements

Now for a look at each of the key syntax elements. The first (and most complicated) one is a "variable." The name must be isolated by looking for the first illegal character or the word THEN. The illegal characters are "+", "-", "<", "=", and ";". The first four are illegal because they are binary operators, and we want "SUM1+SUM2" to be taken as the variable "SUM1", the operator "+", and the variable "SUM2". Semicolon is illegal because we want "INPUT A;B" to be taken as "INPUT", the variable "A", the separator ";", and the variable "B". If semicolon was a legal variable name character it would be taken as "INPUT" and the variable "A;B".

Most BASICs ignore all spaces not in quotes or REM statements, which leads them into such follies as turning MIN THEN into MINTHEN and then into M INT HEN.

THEN must be excluded so that "IF SUM1 THEN" will exclude the THEN from the variable name, thus making the variable "SUM1" rather than "SUM1 THEN". Further, variables are allowed to start only with letters. This is a rather arbitrary restriction, the main use of which is to avoid taking "-5" or "16" as a variable. The programming for all this begins by checking to see if the first character of the variable name is a letter:

```
120 VPNT=0 : IF IN(I)<65 OR IN(I)>90 THEN POP : PRINT "ILLEGAL VARIABLE." : GOTO 5000
```

"VPNT" will point into temporary array T2 where the variable name will be constructed (copied from the input line). It will also be useful for determining the number of characters in the name. Each character must now be checked for illegal characters and THEN.

```
130 A=IN(I) : IF NOT A OR A=43 OR A=45 OR A=60 OR A=61 OR A=59 OR A=84 AND IN(I+1)=72 AND IN(I+2)=69 and IN(I+3)=78 THEN 150
140 T2(VPNT)=A : I=I+1 : VPNT=VPNT+1 : IF VPNT MVAR THEN 130
```

Note that 0 must also be considered an illegal character since it marks the end of the line. The checking process continues until an illegal character or THEN is found, or until the maximum number of characters allowed have been accumulated. If there are no characters in the name (e.g., the name started with THEN), an error must be printed. If not, trailing spaces must be removed.

```
150 IF NOT VPNT THEN IN(I)=0 : GOTO 120
155 IF T2(VPNT-1)=32 THEN VPNT=VPNT-1 : GOTO 155
```

dilithium Press

NEW Educates

ARE YOU COMPUTER LITERATE?

by Karen Billings and David Moursund

Two professional educators with a firm belief in the necessity of computer literacy lead the reader through an enjoyable and informative exploration of the world of the computer. Chapter quizzes and a final exam help the reader chart his or her progress in the self-instruction course.

ISBN 0-918398-20-0 $6.95 paper

INSTANT BASIC

by Jerald R. Brown

Acclaimed by educators and users alike, this easy-to-read, informal approach to DEC's BASIC PLUS language is the painless way to learn BASIC.

ISBN 0-918387-21-5 $9.95 paper

To order these books, or for more information write: dilithium Press, Box 92, Forest Grove, Oregon 97116. Add 50¢ postage & handling for each book you order.

DEALERS: WRITE FOR INFORMATION ON PLACING dP TITLES IN YOUR STORE

CIRCLE 142 ON READER SERVICE CARD

FREE Catalog

New 4-way relief from problems with minicomputer supplies and accessories.

1. One-stop shopping. Inmac (formerly known as Minicomputer Accessories Corporation) has a catalog of over 800 products. Everything from racks and line-printer paper to connectors and cables. Each designed to help keep your minicomputer or word processing system up and running.

2. Hassle-free ordering. Inmac lets you order by mail or phone. So keep our Summer '79 catalog close. It makes those once-tough tasks like ordering magnetic media easy, fast and foolproof.

3. Fast shipment of just the quantity you need. Inmac ships your order within 36 hours from centers in California and New Jersey. In a bind? Call us for many special services that insure you get your order to your installation within 24 hours. Call now and give us a chance.

4. Field-proven quality means precision performance. Inmac guarantees every product in these 70 pages for at least 45 days. And even some for up to ten years.

Inmac
International Minicomputer Accessories Corporation

Send for your FREE Inmac catalog or call (408) 727-1970 today!

2465 Augustine Drive, P.O. Box 4780, Santa Clara, CA 95051

© 1979 International Minicomputer Accessories Corporation

CIRCLE 148 ON READER SERVICE CARD

New North Star Software

DOS + $35.00
Enables any program to execute all North Star Disk and/or Meca Tape commands. Allows batch command list and more.

PRO-TYPE WORD PROCESSOR $75.00
Easy to learn. Combines text input, editing and printing in one program. Features right margin justification, tabs, paging, underlining, relocation of text blocks, etc. Requires only 8K of memory. Manual alone, $25.

GUIDE TO BASEX $33.00
A new interactive compiler similar to BASIC for 8080-type microcomputers (Z-80, 8085). Executes programs up to 10 times faster than equivalent programs while requiring about half the memory space. Features include: array variables; string manipulation; arithmetic operations on signed 16 bit intergers; and versitile 10 communication functions. Manual alone $8.

BASEX TAPE AND DISK GUIDE $35.00
Allows your BASEX programs to access up to four North Star Disk and/or Meca Tape drives. All operations can be executed from the Keyboard. Manual alone $20.

Specify: North Star Disk/Meca Tape/Other
Send for Free Literature

imi

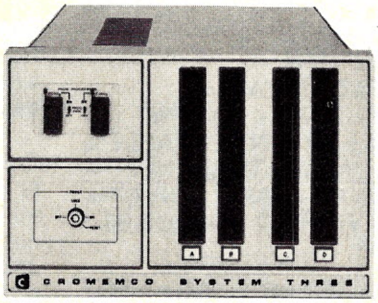
INTERACTIVE MICROWARE INC.
P.O. Box 771, State College, PA 16801
(814) 238-8294

CIRCLE 150 ON READER SERVICE CARD

DISCOUNT PRICES

Microcomputers & Peripherals

**BITS
BYTES
BOOKS
BARGAINS**



**Cromemco • SWTPC • Lear-Siegler
Hazeltine • RCA • North Star
Verbatim • Perkin Elmer and others**

Fast, off the shelf delivery.
Call TOLL FREE 800/523-5355

MARKETLINE SYSTEMS, Inc.
2337 Philmont Ave., Huntingdon Valley, Pa. 19006
215/947-6670 • 800/523-5355

Dealer Inquiries Invited

CIRCLE 158 ON READER SERVICE CARD

Q&A, con't...

Now we must check to see if the variable name is already in TABLE%. The first item in TABLE% is the value of the first variable. The next item is the length of the variable name in characters. This is followed by the variable name, and then all this information repeats for the next variable. TABLE% begins with element 0.

1	+
2	-
3	
4	=

Table 4.

160 FOR A=1 TO NVAR-1 : IF TABLE%(A)< >VPNT THEN 190

If the lengths don't match, it can't be the right name. If they do match, each character must be checked:

170 FOR B=1 TO VPNT : IF TABLE%(A+B) < > T2(B-1) THEN 190

180 NEXT B : N=A+9999 : GOSUB 60 : GOTO 30

When a match is found, the pointer value is stored and the variable subroutine is exited through line 30. The stored value minus 9999 will point to the length of the variable name, and the stored value minus 10000 will point to the value of the variable. If the name does not match, the next name must be checked:

190 A=A+TABLE%(A)+1 : NEXT A

If it isn't in the table, we check to see that there is enough room to add it. If so, it is copied into the table, setting the initial value to zero.

200 IF NVAR+NPNT>MTABLE-2 THEN POP : PRINT "TOO MANY VARIABLES." : GOTO 5000

210 TABLE%(NVAR)=0 : TABLE%(NVAR+1)=VPNT : FOR A=0 TO VPNT-1 : TABLE%(NVAR+A+2)=T2(A) : NEXT A : N=NVAR+10000 : NVAR=NVAR+VPNT+2 : GOSUB 60 : GOTO 30

Originally line 200 read NVAR+NPNT+2>MTABLE THEN which is a little easier to understand, but the end of which becomes MTAB LET HEN. (Note that this cannot happen with the variable subroutine just described, even if a check was being made for LET rather than THEN, because the space between MTABLE and THEN would prevent the formation of LET.)

The next key syntax element is "value." Often this is referred to as an "expression" or an "arithmetic expression" but "value" is much easier to say and type. We begin by checking for "-" (ASCII 45) and the digits (ASCII 48 through 57). If one of these is found, the item must be a number, otherwise it must be a variable.

220 IF IN(I)=45 OR IN(I)>47 AND IN(I)<58 THEN GOSUB 80 : GOSUB 60 : GOTO 240

The subroutine to translate a number already exists at line 80. The number is then stored using the subroutine at line 60. If it is a variable rather than a number, a store-a-variable subroutine is used at line 120:

230 GOSUB 120

Next, for either a number or a variable, a check is made for a binary operator. If there isn't one, the value is finished. Otherwise, the operator is stored and we go back up to find a number or a variable.

240 A=IN(I) : N=(A=43)+2*(A=45)+3*(A=60)+4*(A=61) : IF NOT N THEN RETURN

250 GOSUB 60 : I=I+1 : GOSUB 30 : GOTO 220

One would normally use DATA for the various operators rather than a sum of logicals (in line 240), but Applesoft doesn't seem to have any way to RESTORE data beginning at a certain line, and I didn't want to RESTORE all the data and then read through the syntax data just to get to the operator data.

Variables are allowed to start only with letters. This is a rather arbitrary restriction, the main use of which is to avoid taking "-5" or "16" as a variable.

The next key syntax type is a "variable list." The syntaxing is quite simple and mostly just calls the "variable" subroutine (line 120). Semicolons (ASCII 59) separate the variables and are checked for, but there is no need to store them.

260 GOSUB 120 : IF IN(I)< > 59 THEN 30

270 I=I+1 : GOSUB 30 : GOTO 260

"Value list" is quite similar, but calls "value" (line 220). Semicolons must be stored lest small positive numbers be confused with operators.

280 GOSUB 220 : IF IN(I)< > 59 THEN 30

290 I=I+1 : GOSUB 30 : N=-10001 : GOSUB 60 : GOTO 280

The next syntax element is just the equal sign in the LET statement. It is noted but not stored:

300 IF IN(I)< > 61 THEN POP : PRINT "= EXPECTED." : GOTO 5000

310 I=I+1 : GOTO 30

The next element is "line number" which simply calls the subroutine already present at line 50. (See the ON GOSUB in line 5080.) THEN is next, which is noted and stored as -10001:

320 IF IN(I)< > 84 OR IN(I+1)< > 72 OR IN(I+2)< > 69 OR IN(I+3) < >78 THEN POP : PRINT "THEN EXPECTED." (GOTO 5000 325 I=I+4 : GOSUB 30 : N=-10001 : GOTO 60

Finally, we have "statement." All that is necessary is to forget that a subroutine was called (in line 5080) and start over again at the "statement" routine. Since PNT has been advanced, the statement will automatically be stored after the beginning of the IF statement already stored (since IF is the only statement which uses the syntax element "statement").

330 POP : GOTO 5030

Line Processing

When the line is finally all syntaxed and stored in temporary array T1 as a series of numbers, we're ready to insert it in the proper place within the PROG% array which holds the whole program. T1(0) contains the line number, whether adding a line or deleting one, when we arrive at line 5090. PNT indicates the length of the line (and is 0 when deleting a line). This will be copied into L (for "length") because at one point we will need a variable which indicates the amount of extra space needed, which may be different than PNT if a line is being replaced with a longer line. If there are no lines in the program at the moment, a skip is made down to line 5160 (***). Otherwise, a scan through PROG% is used to find the proper position for the line.

5090 A=0 : PA=-10000 : L=PNT : IF L AND NOT NPROG THEN 5160

5100 FOR A=0 TO NPROG-1 : IF PA=-10000 and PROG%(A)<= T1(0) THEN 5120

5110 PA=PROG%(A) : NEXT A : PROG%(A)=0

The "PA" stuff is to get around the fact that most BASICs don't allow

TRS-80

MEMORY EXPANSION KITS $79

Each Kit consists of: 8 Memory Chips, Jumper Blocks, and Complete EASY TO FOLLOW Instructions
Expands 4K TRS-80 up to 48K (3 sets)
TRS-80FLOPPY Disk (SA-400) Add On
COMPLETE Ready to use with power supply and case $389

ADD to your APPLE or S100 Bus Computer
$89 – Set of 8 250 NS
$99 – Set of 8 200 NS

No. 4116 – 200 NS (w/16K Chips), 16K, $279, 32K, $375, 48K, $469, 64K, 568

No. 4115, 8K, $189, 16K, $229, 24K, $269, 32K, $309

S100 Bus Expandoram Kits*

*Expand NOW or LATER to 64K (32K for *K Chips)
8K Chips: $49/Set of 8
ASSEMBLED, TESTED AND BURNED IN – ADD $50
MONEY BACK GUARANTEE
FULLY WARRANTED FOR 6 MONTHS
Master Charge – VISA – C.O.D. (25% with order) – Money Order – California Residents add 6% Sales Tax
Shipping Charges: $2.00

MicroComputerWorld
P.O. Box 242 San Dimas, CA 91773
(213) 286-2661

CIRCLE 162 ON READER SERVICE CARD

The Best of BYTE

This is a blockbuster of a book containing the majority of material from the first 12 issues of **Byte** magazine. The 146 pages devoted to hardware are crammed full of how-to articles on everything from TV displays to joysticks to cassette interfaces and computer kits. But hardware without software might as well be a boat anchor, so there are 125 pages of software and applications ranging from on-line debuggers to games to a complete small business accounting system. A section on theory examines the how and why behind the circuits and programs, and "opinion" looks at where this explosive new hobby is heading.

Softbound, 386 pages, $11.95 plus $1 shipping and handling in U.S. ($2 foreign).

Send to Creative Computing, P.O. Box 789-M, Morristown, N.J. 07960. Residents of the Garden State add 5% tax. Orders must be prepaid.

For faster service, call in your bankcard order to:

800-631-8112
(In N.J., call 201-540-0445)

creative computing

OMNICOMPUTER PRODUCTS

S-100 • TRS-80 APPLE • PET ADD-ONS

SPEECH SYNTHESIZER

Quality, intelligible natural sounding COMPUTALKER Speech Synthesizer, a proven superior product since 1976, comes complete with Load'N'Go software, user documentation and source programs.

CT-1	S-100 Bus Plug-In	$495
CT-1T	TRS-80 add on w/interconnect to self contained enclosure	595
CT-1A	Apple add on w/interconnect to self contained enclosure	595
CT-1P	Pet add on w/interconnect to self contained enclosure	595

FLEXIBLE DISK DRIVES

Quality single or dual headed single or multi-drive configurations. Interconnects to self contained enclosures. Load'N'Go software and cable interconnects available when required.

35/40 TRACKS

Flip Sided	Dual Sided
40F1 Single Drive $400	40D1 Single Drive $500
40F2 Two Drives $700	40D2 Two Drives $800
40F3 Three Drives $990	40D3 Three Drives $1090

ALL PRODUCTS

Fully Warranted – Sold Direct – Shipped Direct from Stock.
Please contact us for prompt, personal, professional service.

DIAL (714)

NO.1 OMNI

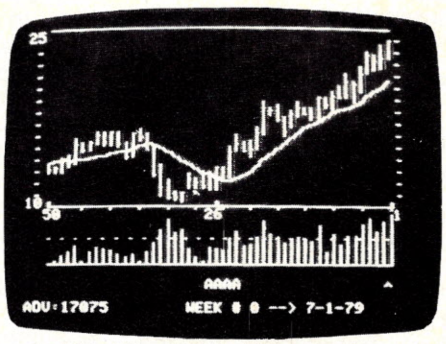
(714) 661-6664

32422 Alipaz St., San Juan Capistrano, CA 92675

CIRCLE 173 ON READER SERVICE CARD

STOCK MARKET SYSTEM

For The Apple II With Disk II



Individual Chart/12 Week Running Average

- INDIVIDUAL CHARTS
- COMPARISON CHARTS IN COLOR
- RUNNING AVERAGES PLOTTED AND ERASED
- RESISTANCE/SUPPORT LINES PLOTTED
- MAINTENANCE PROGRAM MAKES IT EASY TO UPDATE DATA, HANDLE STOCK SPLITS, ETC...
- ONE YEAR'S WORTH OF WEEKLY STOCK DATA (HIGH, LOW, CLOSE & VOLUME) ON YOUR CHOICE OF ANY STOCKS ON THE NYSE OR AMEX
- PROGRAMS & MANUAL........$79.95
  STOCK DATA.................. $9.95 per stock
  MANUAL ONLY................ $4.95

- FOR MORE INFORMATION VISIT YOUR LOCAL COMPUTER STORE OR WRITE DIRECTLY TO:

RTR SOFTWARE INC.
DEPT. CC9
P.O. BOX 12351
EL PASO, TX 79912

CIRCLE 187 ON READER SERVICE CARD

PROG%(A-1) if A=0, and I was foolish enough to actually use element 0 of the array. If I had started using PROG% at element 1, PROG%(0) could have been set to -10000 and then we could change "IF PA=-10000 AND PROG%(A)<=T1(0)" to the more rational "IF PROG%(A-1)=-10000 AND PROG%(A)<=T1(0)" and avoid using PA at all. PROG%(0) need only be set at the very beginning of the program. However, being more used to machine language (where "arrays" are imaginary and you can start the "subscripts" at any number you like), I didn't notice this difficulty until it was well entrenched in the program. It is a very sloppy technique, but it doesn't matter very much so I'm leaving it in, just to give everyone something to sneer at.

If the proper line isn't found at all, then the first unused element in PROG% is set to 0 (in line 5110) to avoid "replacing" a line that isn't there. If a line with a higher line number than the new line is found, or if the end of the program is found, we wind up at line 5120, with A pointing to the higher-numbered line or the first unused element in PROG%.

If the line number at element A in PROG% matches the line number of the new line, a line is to be replaced or deleted. If it doesn't match, then a line is to be inserted or the line to be deleted doesn't exist. An error message is printed if the line doesn't exist:

```
5120 IF PROG%(A)=T1(0) THEN 5170
5130 IF NOT PNT THEN PRINT "NO SUCH LINE." : GOTO 5000
```

Line 5140 is where a line is inserted, and line 5170 will be where a line is replaced or deleted. If a line is to be inserted, then first a check is made to see if there is enough space for it. If so, a "hole" will be expanded into the PROG% array by line 5150 (***), and the line will be copied into this hole (line 5160 ***).

```
5140 IF NPROG+L>MPROG THEN PRINT "PROGRAM TOO LARGE." : GOTO 5000
5150 IF L THEN FOR B=NPROG -1 TO A STEP -1: PROG%(B+L) =PROG%(B) : NEXT B
5160 NPROG=NPROG+L : QNVAR=NVAR : IF PNT THEN FOR B=0 TO PNT-1 : PROG%(A+B)=T1(B) : NEXT B
5165 GOTO 5000
```

A somewhat obscure item in line 5160 is the "QNVAR=NVAR". A similar obscurity occured way back at line 5000 ("NVAR=QNVAR"). This is mainly due to a problem in BASIC called the "implied LET" statement. I had originally intended to name QNVAR PNVAR (for "previous NVAR") but that conflicts with PNT

Now all that remains is to execute the three commands. If the first letter of the input line is an L, then it must be the list command.

(grumble). When a syntax error occurs, NVAR will be set back to the previous NVAR (QNVAR) by line 5000. This deletes all variables added to TABLE% during processing of the illegal line. However, when a valid line is entered, line 5160 sets QNVAR equal to NVAR so line 5000 won't change the value of NVAR (thus keeping all the variables added to TABLE%). What does all this have to do with the LET statement? Well, suppose I type in 10 GOSUB 50. There isn't any GOSUB in this language, therefore, since GOSUB doesn't match any of the keywords, a LET statement is assumed. GOSUB 50 is a legal variable name so it is added to the TABLE%. However, since "=" is expected but not found, "= EXPECTED." message will be printed. (This is rather cryptic for an error message, but it is what the syntaxer is "thinking." You can jazz the errors up quite a bit just by adding a PRINT LEFT$(A$,I+1); in front, like PRINT LEFT$(A$,I+1);" = EXPECTED.". This can be done to all the syntax errors. In this case, it would print "10 GOSUB 50 = EXPECTED.", and if you fill in the equal by typing 10 GOSUB 50=, then it would print "10 GOSUB 50 = ILLEGAL VARIABLE." assuming you fixed up the other messages. Then you just supply a legal variable like: 10 GOSUB 50 = SUM1, and (presto!) you get a legal statement. It will probably begin to dawn on you that it thinks "GOSUB 50" is a variable.) It would be crude to fill the TABLE% up with all these screwball variable names, so the simple QNVAR trick prevents it.

To get back to finished line processing, line 5170 is the place where lines are replaced or deleted. The old line, or the line to be deleted, is pointed to by A. It is now necessary to determine the length of the old line. Usually people begin each line by storing the length, for a variety of good reasons. In this case I didn't.

```
5170 FOR B=1 TO NPROG : IF PROG%(A+B-1) < > -10000 THEN NEXT B
```

The length of the old line is now in B. If the new line is larger than the old line (or the same size), we can use the routine already written for "inserting" a Line must be set to the difference in lengths and it will then make a hole big enough for the increased size.

```
5180 IF B<=PNT THEN L=PNT -B : GOTO 5140
```

Line 5140 will even check to make sure there is enough room to add the longer line. If the new line is shorter than the old, or if a line is being deleted we have to squeeze PROG% down to eliminate the difference in size (or the whole line). Then we can use line 5160 to copy the new line in (if any).

```
5190 FOR L=A+B TO NPROG-1 : PROG%(L-B+PNT)= PROG%(L) : NEXT L : L=PNT-B : GOTO 5160
```

L must be set to the difference in length so NPROG w be correctly updated.

More Housekeeping

That's it for syntaxing! We now have a program which will accept statements, grind them down to numbers and store them away in the proper order. Now we need a couple of subroutines to convert line numbers from numbers to pointers and vice-versa. The subroutine to convert numbers into pointers looks like this:

```
410 PA=-10000+NOT NPROG : FOR A=0 TO NPROG-1 : IF PA<> -10000 THEN 500
415 L=A
420 A=A+1 : IF PROG%(A)=4 THEN IF PROG%(A+1) 0 THEN 460
430 IF PROG%(A)<>5 THEN 500
440 A=A+1 : IF PROG%(A)< > -10001 THEN 440
450 GOTO 420
460 C=PROG%(A+1) : PA= -10000 : FOR B=0 TO NPROG-1 : IF PA=-10000 AND PROG%(B) =C THEN 490
470 PA=PROG%(B) ( NEXT B : IF RFLAG=1 THEN POP : PRINT "UNDEFINED REFERENCE IN LINE";-PROG%(L); "." : GOTO 5000
480 GOTO 500
490 PROG%(A+1)=B
500 PA=PROG%(A) : NEXT A : RETURN
```

NEW!

Apple® software from RAINBOW

PIE TEXT EDITOR Machine language, cursor-based text editor for 16K Apple.

- Features format capabilities of most text editors.
- All commands are control characters.
- Enables you to define your own function commands.

Order PIE on cassette: $19.95. . . . . .on diskette $24.95

HIGH RESOLUTION CHARACTER GENERATOR Machine language program for 16K Apple.

- Define your own character set and graphic shapes.
- Complete English upper/lower case character set.
- Complete Greek Alphabet with upper/lower character set.
- Scroll, vary window size, invert characters, switch back and forth between two character sets.

Order Hi-Res Char. Gen. on Diskette $19.95

FORTE Music Interpreter in Machine Language for 16K Apple.

- **Handles six voices.**
- **Single step capability.**
- **Full editing features.**
- **Trace line numbers or notes.**
- **Save songs on cassette or diskette.**

Order FORTE on Cassette $19.95

APPLE Monitor PEELED Everything you wanted to know about the Apple Monitor but couldn't figure out. User-written manual in plain English clears your confusion **$9.95**

Call or write today for your FREE Apple Software Catalog. B/A and Mastercharge accepted. Sorry, no CODs. Add $1.25 Shipping & Handling. California residents add 6% sales tax. We ship promptly on receipt of your ppd. order. Order from:

RAINBOW COMPUTING INC.
Garden Plaza Shopping Center, Dept. 8CC
9719 Reseda Blvd., Northridge, CA 91324
Telephone (213) 349-5560

tape'n text

TAPE 'N TEXT IS THE NAME TO BUY FOR THE VERY BEST IN AUDIO TUTORIAL LEARNING

The best selling Tape 'N Text learning aids are being used in over 700 schools, colleges and industries:

BASIC Programming Course 12 audio cassette tapes and 12 printed texts **$59.85**
Introduction to PASCAL 4 audio cassette tapes and 4 printed texts **$24.95**
Microcomputers For Businessmen 4 audio cassette tapes and 4 printed texts **$19.95**
Basic Math For Computers 4 audio cassette tapes and 4 printed texts **$19.95**

Also, we announce the following new Apple II computer application software. Requires: Apple II computer - 32K RAM, One Drive, Disk II, Applesoft II in ROM - firmware card:

Disk A1 - **Book Library** - over 1000 cross-referenced titles may be stored - title, subject and author. **$19.95**

Disk A2 - **Record Library** - over 1000 cross-referenced titles may be stored - title, composer, and performer. **$19.95**

Disk A3 - **Malum II** - A Latin language based interpreter similar to BASIC "by command of Caesar". **$19.95**

Disk A4 - **Graphics Games** - Bowling, Checkers, The Derby, and Tic-Tac-Toe. **$19.95**

Disk A5 - **Checkbook Program** - can store up to 66 checks and deposits per month (32K); 250 checks (48K). **$19.95**

The new programs were written by John H. Barnes and edited by William R. Parks, Assistant Professor, State University of New York. The audio cassette courses were written by William R. Parks.

Add $1.50 for shipping (Overseas add $5). N.Y.S. residents add 7% sales tax. Make checks payable to: **Williamsville Publishing Company**, Dept. CC-1, P.O. Box 237, Williamsville, N.Y. 14221.

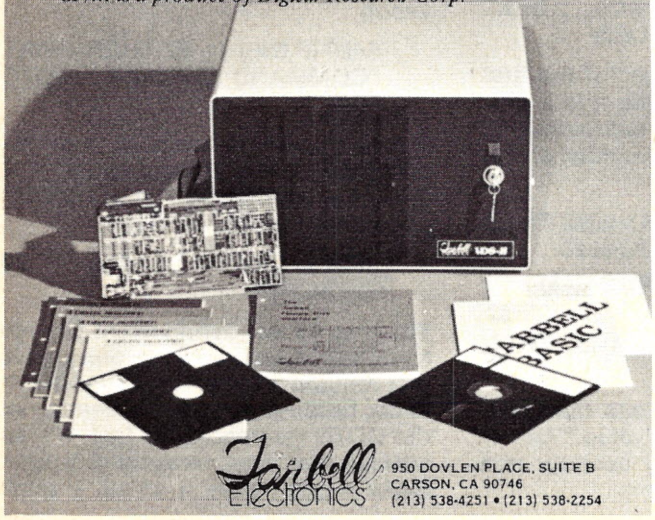
TARBELL VDS-II
Vertical Disk Subsystem

SYSTEM INCLUDES:

- 2 Siemens 8" Disk Drives
- 1 Cabinet with Fan and Power Supply.
- 1 Tarbell Floppy Disk Interface, assembled & tested.
- 1 CP/M* Disk Operating System.
- 1 Tarbell BASIC.
- All Cables and Connectors.
- Complete User Documentation.
- Fully Factory Assembled and Tested.

Price $1888.00

CP/M is a product of Digital Research Corp.



Tarbell Electronics
950 DOVLEN PLACE, SUITE B
CARSON, CA 90746
(213) 538-4251 • (213) 538-2254

TORA SYSTEMS INC.

COMPLETE BUSINESS SYSTEMS $3498.27

32K TRS-80 + 779-2 Tractor Feed + 4 Basf Drives

Level II-16K (w/o num. keypad)	$799.00
Expansion Interface Ø	$269.00
Expansion Interface 16K	$423.00
Expansion Interface 32K	$592.00
16K Memory Expansion Kit (with Jumpers and inst)	$ 89.95

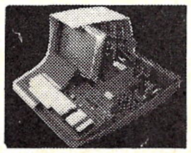
***** MAIL ORDER ONLY *****

VERBATIM FLOPPY MINI DISKS
BOX OF 10, ONLY $3.20 PER DISK
DISK DRIVES . . INCLUDING TRS-80

ADD-ON

SA-400 SHUGART DISK DRIVE FOR TRS-80 Includes power supply and custom walnut formica enclosure (see below)	$ 415.00
SA-400 SHUGART DISK DRIVE ONLY	$ 315.00
6106-BASF 5-1/4" MINI FLOPPY DRIVE for TRS-80 Includes power supply and custom walnut formica enclosure (see below)	$ 395.00
BASF DISK DRIVE ONLY	$ 299.00
SEAMAN DRIVE	$ 425.00
HAZELTINE 1400 WHILE SUPPLY LASTS	$ 649.49
HORIZON I KIT	$1339.00
CENTRONICS 799 PRINTER-TRACTOR (TRS-80)	$1099.00
ENCLOSURE FOR SHUGART SA-400 WALNUT FORMICA CUSTOM DESIGN	$ 27.95
ENCLOSURE FOR DUAL SHUGART SA-400	$ 39.95
RIBBONS FOR MOST PRINTERS	INQUIRE

INTERTUBE TERMINAL
$784.00



WE ACCEPT BANK AMERICARD, VISA, MASTER CHARGE

29-02 23RD AVENUE ASTORIA, NEW YORK 11105
212-728-5252

Q&A, con't...

The + NOT NPROG in line 410 checks for a null program (***). Line 420 checks for statement type 4, which is GOTO (the only statement which can have a line number reference). If found, line 460 changes the number to a pointer reference, if it wasn't one already. Line 430 checks for statement type 5 (IF) since it contains a statement which could be a GOTO (or another IF). One obscure item is the check of RFLAG in line 470. For now, it is sufficient to know that the RUN command will set RFLAG to 1.

The subroutine which will undo all of this looks like:

```
340 PA = -10000 + NOT NPROG :
FOR A = 0 TO NPROG-1 : IF PA<>
-10000 THEN 400
350 A = A + 1 : IF PROG%(A) = 4
THEN IF PROG%(A + 1)>= 0
THEN 390
360 IF PROG%(A)<>5 THEN 400
370 A = A + 1 : IF PROG%(A)<>
-10001 THEN 370
380 GOTO 350
390 PROG%(A + 1) = PROG%
(PROG%(A + 1))
```

Everyone knows that renumber is particularly difficult to do, because you have to change all the GOTO references. In fact, it's so difficult that most personal computer BASICs don't have it at all.

```
400 PA = PROG%(A) : NEXT A :
RFLAG = 0 : RETURN
```

Again (line 340), + NOT NPROG checks for a null program (***). You may recall that this subroutine was called way back in line 5010. Line 400 sets RFLAG to 0 so this routine won't be called by line 5010 again until it is necessary.

Now all that remains is to execute the three commands. If the first letter of the input line is an L, then it must be the list command. (This command syntax checker is pretty crude, but it is not significant to this example.) If there is no program to list, then line 5205 goes back to the line input routine (***).

```
5200 IF IN(I)<>76 THEN 5420
5205 IF NOT NPROG THEN 5000
```

(Line 5420 is where we wind up if the first letter isn't L.) For each line, the line number must be printed:

```
5210 PRINT : FOR A=0 TO
NPROG-1
5215 PRINT -PROG%(A);" "; :
A=A+1
```

Now the data is run through (the same data used in syntaxing an input line), search for a matching statement type number and print the keyword.

```
5220 RESTORE
5230 READ L : FOR B = 1 TO L :
READ T1(B) : NEXT B : READ B :
IF B = PROG%(A) THEN 5260
5240 READ L : IF L<> 255 AND L
THEN 5240
5250 IF L THEN 5230
5255 READ B
```

The statement keyword has been found by the time we get to line 5260. It has been stored in temporary array T1. L is the length of the keyword (or zero for LET), and the next data item to be read is the syntax of the statement. First, the keyword (***) is printed:

```
5260 IF L THEN FOR B = 1 TO L :
PRINT CHR$(T1(B)); : NEXT B :
PRINT " ";
5265 A = A + 1
```

Now we simply read the syntax out of the data, and list the rest of the line. An ON GOSUB is used to call a subroutine for each syntax type.

```
5270 READ L : IF L< > 255 THEN
ON L GOSUB 5290,5300,5340,
5360,5380,5390,5410,5415 :
GOTO 5000
```

Line 5280 is used when the whole line is finished. Now for a look at each syntax item.

The first syntax item is "variable". A variable (L) is set to the location of the length and name, then the name is printed.

```
5290 L = PROG%(A)-9999 : FOR
B=1 TO TABLE%(L) : PRINT
CHR$(TABLE%(B + L)); : NEXT
B : A = A + 1 : RETURN
```

Next, we have "value." If the first item is greater than 9999, it is a variable and the above subroutine is used. Otherwise, it is a number and is just printed.

```
5300 IF PROG%(A)>9999 THEN
GOSUB 5290 : GOTO 5320
5310 PRINT PROG%(A); :
A=A+1
5320 IF PROG%(A)<0 THEN
RETURN
```

At line 5320, a value less than zero means the end of the "value." Otherwise, there is a binary operator and the whole thing starts over.

```
5330 A$ = " + -<= " : PRINT MID$
(A$,PROG%(A),1); : A = A + 1 :
GOTO 5300
```

Next, we have "variable list" which is rather trivial:

```
5340 GOSUB 5290 : IF PROG%
(A)<>-10000 THEN PRINT ";"; :
GOTO 5340
5350 RETURN
```

Similarly, for "value list" the values are printed separated by semicolons until the end of line marker (-10000) is reached.

```
5360 GOSUB 5300 : IF PROG%(A)
< > -10000 THEN PRINT ";"; :
A = A + 1 : GOTO 5360
5370 RETURN
```

Here's a real tricky one for the LET statement's equal sign:

```
5380 PRINT " = "; : RETURN
```

The next item is "line number", and we must take into account that line numbers are stored either as negative numbers or as non-negative addresses. We could have just done GOSUB 340 to change them all to negative numbers but that would be crude.

```
5390 IF PROG%(A)<0 THEN
PRINT -PROG%(A); : A = A + 1 :
RETURN
5400 PRINT -PROG%(PROG%
(A)); : A = A + 1 : RETURN
```

Next, we have another devious subroutine:

```
5410 A = A + 1 : PRINT " THEN ";
: RETURN
```

Finally, we do "statement" in the usual manner:

```
5415 POP : GOTO 5220
```

Line 5420 is where we wind up if a command is typed that doesn't start with the letter L. If it doesn't start with R either, then there is something wrong.

```
5420 IF IN(I)< > 82 THEN PRINT
"COMMAND ERROR." : GOTO
5000
```

Now we have a choice of RUN or RENUMBER. If the second letter is E then the command is renumber, otherwise we'll branch to run.

```
5430 A = IN(I + 1; : IF A< > 69
THEN 5460
```

You may be wondering why I choose to include **renumber**. Everyone knows that renumber is particularly difficult to do, because you have to change all the GOTO references. In fact, it's so difficult that most personal computer BASICs don't have it at all. I think

Q&A, con't...

we'll just do it an easy way to spite them all. First, we'll call the same routine run will use to turn line numbers into pointers then just change the line numbers and forget about the GOTO references.

```
5440 RFLAG = 2 : GOSUB 410 : L = -10 : PA = -10000 : FOR A = 0 TO NPROG-1 : IF PA = -10000 THEN PROG%(A) = L : L = L-10
5450 PA = PROG%(A) : NEXT A : GOTO 5000
```

Setting RFLAG to 2 causes the 410 subroutine to just ignore any undefined references and leave them as negative numbers (when called by run with RFLAG = 1, an error message will be printed for undefined references). Since the GOTO line numbers just point to the real line numbers and we've just changed all those, the line numbers in the GOTO statements will be changed without having to bother with them. Undefined references will be left alone, which is preferable. (The above routine numbers by 10's starting with 10, but could be modified for more complex numbering.) Clean living has its benefits.

Now at line 5460 we have the run command. It looks like this:

```
5460 IF A<>85 THEN IN(I) = 0 : GOTO 5420
5470 RFLAG = 1 : GOSUB 410 : PNT = 0 : GOTO 620
```

Line 5460 makes sure the second letter of the command is a U. For obvious reasons we have put most of the run command near the beginning of the program.

At line 620 the pointer (PNT) has been copied to the current line number into LINE for future reference. If there are no more lines to run a branch is made to the line input section (line 5000).

```
620 LINE = PNT : PNT = PNT + 2 : IF LINE = NPROG THEN 5000
```

PNT is advanced by 2 so as to point to the first parameter (past the statement type number). An ON GOTO statement will be used to branch to the appropriate statement's routine:

```
630 ON PROG%(PNT-1) GOTO 670,680,740,760
```

The IF statement must determine the value of a given "value." A subroutine (beginning at line 510) to evaluate a value and return the answer in N will be used.

```
510 GOSUB 590 : N = A
```

Line 590 will return the value of a number or variable in A.

```
520 B = PROG%(PNT) : IF B<0 THEN RETURN
```

The value is complete if no binary operator follows. Otherwise, the second number/variable is evaluated and an ON GOSUB goes to the proper arithmetic routine.

```
530 PNT = PNT + 1 : GOSUB 590 : ON B GOSUB 550,560,570,580 : IF ABS(N)<10000 THEN 520
540 POP : PRINT "OVERFLOW IN LINE";-PROG%(LINE);"." : GOTO 5000
550 N = N + A : RETURN
560 N = N-1 : RETURN
570 N = N<A : RETURN
580 N = N = A : RETURN
590 A = PROG%(PNT) : PNT = PNT + 1 : IF A<10000 THEN RETURN
600 A = TABLE%(A-10000) : RETURN
```

Once the IF statement value has been determined we continue with the statement that follows "THEN" if the value is non-zero:

```
640 GOSUB 510 : PNT = PNT + 2 : IF N THEN 630
```

Otherwise, skip down to the next line:

```
650 PNT = PNT + 1 : IF PROG%(PNT-1)<>-10000 THEN 650
660 GOTO 620
```

The LET statement is quite simple. L is the pointer to the destination variable.

```
670 L = PROG%(PNT)-10000 : PNT = PNT + 1 : GOSUB 510 : TABLE%(L) = N : GOTO 610
```

Line 610 is a general purpose line that increments PNT and then falls through to line 620:

```
610 PNT = PNT + 1
```

The input statement reads in a line (using the subroutine at line 110). If the first letter of the input is an S, program execution is stopped and we branch to the line input section (line 5000).

```
680 PRINT "?"; : GOSUB 110 : IF IN(I) = 83 THEN PRINT "STOPPED AT LINE";-PROG%(LINE);"." : GOTO 5000
```

Next, we check to see if there are any letters left in the input line. If not, go back up and get some more.

```
690 IF NOT IN(I) THEN 680
```

If the first character is not a "-" (ASCII 45) or a digit (ASCII 48 through 57), an error message is printed and another line is asked for.

```
700 IF IN(I) <> 45 AND (IN(I)<48 OR IN(I)>57) THEN PRINT "BAD INPUT." : GOTO 680
```

Otherwise, we translate the number (line 80) and store it in the proper variable. If there are no more variables to input, we're done (branch to line 610).

```
710 GOSUB 80 : TABLE%(PROG%(PNT)-10000) = N : PNT = PNT + 1 : IF PROG%(PNT) = -10000 THEN 610
```

Otherwise, a check is made for a semicolon in the input line, and we go do the next variable. (Semicolons are used in the input string to separate numeric values. I would prefer commas, but they are not allowed in Applesoft.)

```
720 IF IN(I) <> 59 THEN IN(I) = 0 : GOTO 700
730 I = I + 1 : GOTO 690
```

The print statement is quite simple:

```
740 GOSUB 510 : PRINT N;""; : IF PROG%(PNT) = -10000 THEN PRINT : GOTO 610
750 PNT = PNT + 1 : GOTO 740
```

The GOTO statement is also simple:

```
760 PNT = PROG%(PNT) : GOTO 620
```

Conclusion

That's it. In real life it is much more complicated, of course, but this interpreter is intended only to show some basic concepts. You can type the interpreter in and try it if you like, but it is really only to show the exact process. (I've tested it, of course.) If you do run it, you'll notice that it takes quite a while to syntax a line when it is typed in. Keep in mind that a well written interpreter (written in machine language rather than BASIC) can easily do this in a very small fraction of a second. This will put the speed of the RUN command in proper perspective.

Do you have questions related to microcomputers? Send them to this address:

Questions & Answers
Creative Computing Magazine
P.O. Box 789-M
Morristown, N.J. 07960

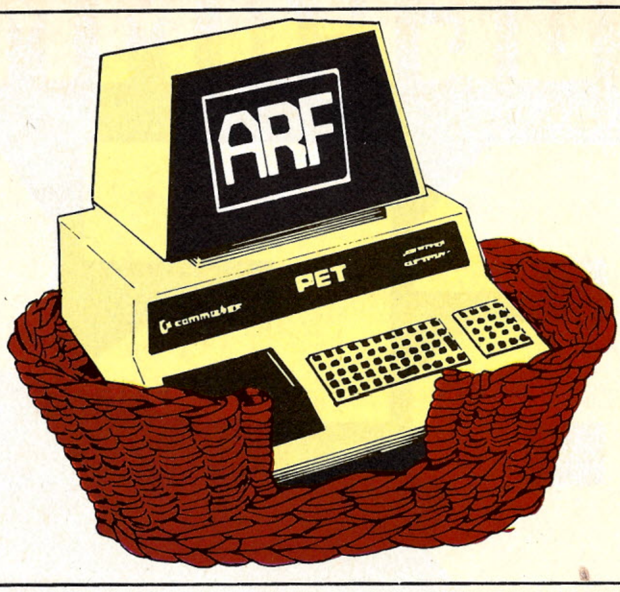
Please try to keep questions general, like "What is an RS-232 printer interface?" rather than, "How do I connect a Cray-30 terminal into my Hyperdata 80 CPU?" □


PROGRAMMING THE
6502
PROGRAMMING THE
6502
SYBEX
RODNAY ZAKS

Personal Electronic Transactions

by Gregory Yob

I am happy to hear from you, and encourage your correspondence. I will try to acknowledge all correspondence, and a SASE makes things easier for both of us. Please send your letters to "Personal Electronic Transactions" c/o PO Box 354, Palo Alto, CA 94301.



Three Things I Can't Ignore

In the sporadic flow of product descriptions that cross my desk, three items appeared that are worth your attention:

1) The BASIC PROGRAMMER'S TOOLKIT (Palo Alto IC's, 810 Garland Drive, Palo Alto, CA 94303) this is a little PC card with some ROM that attaches to your memory expansion port — and adds some **very helpful** commands to your PET's BASIC—especially if you are a programmer! Here is a summary of the Toolkit's features:

AUTO: This will automatically generate the line numbers as you enter program lines. APPEND: You can take a program SAVEd previously and add it to the end of the program currently in memory. (It is up to you to keep the line numbers in order.) DELETE: Delete lines with the same rules as used in LIST. DUMP: Display the variables and values in BASIC's variables area during or after a program RUN. HELP: After an error, the offending line is shown, with the point-of-error indicated. RENUMBER: Renumbers your program. TRACE: Shows in the upper right corner the line numbers being executed. SINGLE STEP: Execution of single statements by pressing the space key (useful with TRACE activated). This little goodie costs $75.00 and will be available when the column is printed (now for you!). Mention that you saw it in this column if you place an order.

2) Micro Technology Unlimited, 841 Galaxy Way, Manchester, NH 03108 offers a graphics board for the PET for $243.00 plus $79.00 for the PET interface. Your PET display now looks like 8K of memory with dot-by-dot addressability! (All 64,000 dots.) MTU offers several other products of interest, so ask for their catalog.

3) There are two vendors offering ways to protect tapes with PET programs on them. Their claim is that the tapes can be duplicated via bulk processing and **won't** let you copy the program once loaded into your PET - this presumably solves the "ripoff" problem for PET software. The suppliers are: A) G.E. Enterprises, 1417 11th Street, Manhattan Beach, CA 90266. B) BC Computing, 2124 Colorado Ave., Sun Prairie, WI 53590. I took a look at the GE Enterprises sample, and it worked to specification.

Though these services are certainly a "good thing" and all of that, please consider the disadvantages. First, a user who buys a protected tape will not be able to copy it - and when the tape wears out after 30-50 LOADs, he will have to request another tape - from you! Second, if you need to update a protected program, you will have to submit the revised version to the protection service and be charged the protection fee. Third, I've noticed that most PET users are not copying tapes indiscriminately - the software "ripoff" hasn't deterred the development of an active PET software market - and at least two suppliers are grossing over $1 million yearly.

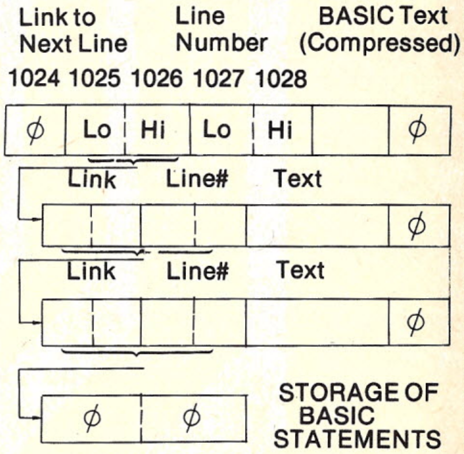
A Random Hint

When I covered the RND function in a previous column, I forgot to mention a useful trick. To make sure the RND seed is reset when a game starts, use a statement like: 100 X = RND(-TI). Since the time in jiffies will vary when you LOAD and RUN a program, you can avoid the problem of turning on your PET, loading a poker game, and always getting the same hand the first time.

On The Structure of BASIC

If you are a beginner, skip on to the next part - this section gets a little technical.

PET's BASIC has to store the program text somewhere in the memory and with PEEK and POKE you can look at a BASIC program, and with some care, make changes. The diagram below shows how PET's BASIC is stored in memory:



STORAGE OF BASIC STATEMENTS

The BASIC program starts at byte 1024, which is always a zero. (If you change this one, you will get into trouble. Try it and see!) The program proper starts at byte 1025.

Each line of the BASIC program has the first four bytes dedicated to the linkage pointer and the line number. At the end of each line is the value zero. The linkage pointer in the last line points to a pair of zeroes, which indicates the end of the BASIC program. (This gives a "triple zero" due to the zero at the end of the last line.)

The link is in the form (low bits) (high bits) and points to the first byte in the next linkage pointer. For example, PEEK (1025) + 256 * PEEK (1026) will show where the second line begins.

The next two bytes are the line number, stored in the same (low bits) (high bits) form in 16 bit binary. Though this permits a line number of 65535, PET's BASIC will not accept line numbers over 63999.

In a previous column I mentioned that POKE 1026,3:LIST would do strange things. Some thought reveals that this makes the first linkage pointer point into lower RAM where the contents aren't BASIC. The PET's lister is rather dumb and proceeds to list the RAM (and if a randomly chosen pointer does so, the ROM too) as a BASIC program. If one "pointer" points back to another "pointer," the lister never finishes and you have to reset the PET.

The text portion holds the program text in a tokenized form. The BASIC keywords, like PRINT, SIN, COS, FOR, etc, are replaced by one byte "tokens." I'm sure you have noticed that graphics characters can be put in a BASIC program only as characters inside quote marks (with the exception of REM).

Let's back this theory up with a few examples. First, enter

```
10 REM A S D F G H J K L :
```

and then do a LIST:

```
10 REM ATNLETSTR$ASCCHR$
LEFT$MID$
?SYNTAX ERROR
```

If you fool around a bit, you will learn that SHIFT-A is the token for ATN, SHIFT-S is the token for LET and so on. One token, SHIFT-K, produces a ?SYNTAX ERROR when LISTed - this is another famous PET oddity. A few of the graphics characters aren't used as tokens in BASIC. Others serve for numbers, operators, etc. (i.e., the operator * is a token which is different from the character * in a string).

Now let's try a small program, and see how it looks inside:

```
10 REM HELLO
FORJ = 1024TO1044:?J,PEEK(J)
 :NEXT
1024    0
1025    13
1026    4
1027    10
1028    0
1029    143
1030    32
1031    72
1032    69
1033    76
1034    76
1035    79
1036    0
1037    0
1038    0
1039    ----
1040    ----
1041    ----    These will vary.
1042    ----
1043    ----
1044    ----
```

Location 1024 has the required zero, as expected. 1025 and 1026 hold 13 and 4 - this works out to a pointer value of 1037. If you look at 1037 and 1038, you will see the double zero which indicates the end of this short program.

Locations 1027 and 1028 hold the line number (10 + 256*0) which is 10. Location 1029 holds the token for REM, which is 143. After that is the string spHELLO. If you want to see this, try:

```
FORJ = 1024TO1044:?J,CHR$
(PEEK(J)):NEXT
```

and the HELLO will appear from 1031 to 1035. (The J at 1040 is the loop counter in variable storage.) The gap between 1025 and 1026 comes from the 13 which is the character for RETURN.

Now, let's have some fun. First, try POKE 1028,255. Now try a LIST:

```
65290 REM HELLO
```

That's a puzzle for you - and how to make lines with illegal line numbers - which can't be deleted, by the way. (Hint: 65290 is 255*256 + 10)

For some more fun, add this POKE:

```
POKE 1025,1
```

And now do a LIST - and 65290 REM HELLO will list forever on your screen (until you press the STOP key). The BASIC LISTer looks at the linkage pointer, lists the following line# and text, and then uses the linkage pointer to find the next line - which has been made to point to itself!

As an exercise, let's build a program that will LIST itself without using the PET's LIST command. For starters, the program will display the linkage pointers and line numbers for one line at a time. Pressing any key will move to the next line. (I am starting at Line 63000 to let this program be added to other programs for the dedicated hacker's use.)

```
SELF-LIST PROGRAM
63000 REM SELF-LISTING
      PROGRAM
63010 REM BY GREGORY YOB
63020 DEF FNF (X) = PEEK(X) +
      256*PEEK(X + 1)
63030 REM LP IS LINK POINTER
63040 REN LN IS LINE NUMBER
63050 LP = 1025
63500 REM DO AUTO-LIST
63510 PRINT"clr sp AUTO-LIST"
63520 PRINT"dn sp LINK sp sp
      LINE# sp sp TEXT";PRINT
63530 GET A$:IFA$ = ""THEN
      63530
63540 LN = FNF(LP + 2):
      LP = FNF(LP)
63550 PRINT LP;TAB(6);LN
63560 IF LP   0 THEN 63530
```

Pet, con't...

When this program is RUN, it clears the screen and prints the title (lines 63510 and 63520). When you press any key, the link and line number for one line is displayed.

The function FNF is very handy when diddling with BASIC, for it converts a pair of (low) (high) bytes to the number they represent. (Note: FNF can be useful to machine language coders too...)

The "guts" of this program is line 63540, which fetches the line number and the linkage pointer.

The next thing to do is to display the BASIC text for each line. First. let's just take a look at the raw text - and later tackle the problem of converting the tokens. Here are the lines to add to the Self-List Program:

```
63060 REM TP IS START OF
      TEXT
63070 REM CH IS TEXT CHAR
63540 LN = FNF(LP + 2):
      TP = LP + 4 : LP = FNF(LP)
63550 PRINT LP;TAB(6);LN;
      TAB(12);
63560 CH = PEEK(TP):IF CH = 0
      THEN 63600
63570 PRINT CHR$(CH);:
      TP = TP + 1 : GOTO63560
63600 PRINT:IFLP    0 THEN
      63560
```

This modification scans the text and prints the text character until a zero is seen. CH is the text character, TP the pointer to the character.

When you RUN this time, notice that the token for REM is a non-printing character. Also notice that the operators + - * / are converted into tokens.

The big challenge is to convert the tokens into their BASIC keywords. This requires storing a table of keywords in a string array whose indice is the token's value (e.g., S$(143) would be REM). The additions to the Self-List Program are:

```
63080 REM S$ HOLDS TOKENS
63090 DIM S$( 75):FORJ = 0 TO
      75: READ S$(J):NEXT
63200 DATA END,FOR,NEXT,
      DATA,INPUT#,INPUT,
      DIM,READ,LET,GOTO
63210 DATA RUN,IF,RESTORE,
      GOSUB,RETURN,REM,
      STOP, ON, WAIT
63220 DATA LOAD,SAVE,
      VERIFY,DEF,POKE,
      PRINT#;PRINT,CONT
63230 DATA LIST,CLR,CMD,
      SYS,OPEN,CLOSE,GET,
      NEW,TAB(,TO,FN
63240 DATA SPC(,THEN,NOT,
      STEP, + ,-,*,/,↑,AND,OR,
      V, = ,V
63250 DATA SGN,INT,ABS,USR,
      FRE,POS,SQR,RND,LOG,
      EXP,COS,SIN
63260 DATA TAN,ATN,PEEK,
      LEN,STR$,VAL,ASC,
      CHR$,LEFT$,RIGHT$
63270 DATA MID$,!!OOPS!!
63560 CH = PEEK(TP):IF CH = 0
      THEN 63700
63570 TP = TP + 1
63580 IF CH > 128 THEN PRINT
      CHR$(CH);:GOTO 63560
63590 IF CH > 203 THEN
      CH = 203
63600 PRINT S$(CH-128);:
      GOTO 63560
63700 PRINT:IF LP > 0 THEN
      63530
```

Now the BASIC program will have the text listed as well - and we are nearly done. Take a look at the listing for line 63510 - the clear screen character is translated to LOAD! When characters inside quotation marks are seen, they **aren't** tokens. A counter of quote marks, QQ, solves this problem:

```
63575 IF CH = 34 THEN QQ = QQ
      + 1AND 1
63555 QQ = 0
63585 IF QQ THEN PRINT CHR$
      (CH);:GOTO 63560
```

Each new line sets QQ to zero. When the quote mark is seen (34) QQ is incremented, and the least significant bit extracted with the AND 1. This "toggles" QQ in the same way that the PET keeps track of "quote mode." When an odd number of quotes is seen, QQ is 1, and line 63585 prints the character.

It is up to you to utilize your knowledge of BASIC's structure. As exercises, consider the problems of renumbering and program appending. Of course there are products in software and firmware that do this in machine language (like the Programmer's Toolkit), so your efforts should be done for learning's sake.

Some Cryptic Things

If you are into secret messages, the PET is an excellent aid for making and receiving encrypted messages. (I will avoid the more difficult art of cryptanalysis.)

For starters, here is the code generated by shifting the alphabet a little:

Cleartext: ABCDEFGHIJKLMNOP QRSTUVWXYZ

Ciphertext: HIJKLMNOPQRSTUVW XYZABCDEFG

The PET alphabetic and numeric characters have the values 32 to 95. By adding or subtracting the number of spaces to shift, it is possible to encode a letter. The example above encodes by adding 7 to the cleartext value, and decodes by subtracting 7. Here is the PET's version:

```
10 PRINT"clr PET CIPHER PRO-
   GRAM"
20 INPUT"dn dn SECRET NUM-
   BERS";S
30 IF S < -64 or S > THEN 20
40 PRINT"clr ENTER YOUR
   MESSAGE"
```

Pet, con't...

```
50 PRINT: INPUT"dn"; M$
60 FOR J = 1 TO LEN(M$)
70 C$ = MID$(M$,J,1)
80 GOSUB 1000
90 PRINT C$;
100 NEXT J
120 GOTO 50
1000 REM CRYPT ROUTINE
1010 C = ASC(C$) + S
1020 IF C < 32 THEN C = C + 64
1030 IF C > 95 THEN C = C-64
1040 C$ = CHR$(C): RETURN
```

When you RUN this, the screen will clear and ask you for the secret number. Then the screen clears again and you enter your message. The PET then prints the encoded version:

```
RUN
(screen clears)
PET CIPHER PROGRAM
SECRET NUMBER? 6
(screen clears)
ENTER YOUR MESSAGE
? HELLO OUT THERE
NKRRU&U&U[Z&ZNKXK
```

hello / o u / there

If you RUN again, with the secret number of -6, and enter the NKRRU... the original message will reappear.

For practical uses, you should add the ability to read or write cassette tapes with your messages on them. I leave this one up to you, with the warning to study how data tapes work first. (There's a tutorial article in the January issues of Kilobaud on this.)

See if you can determine the secret number for this message, and decode it if you can. (Hint: Write a program that tries all the 64 possible secret numbers - one message will make sense. Another hint: See if you can figure out the character for space...)

```
/,336G0.69GTG; /,G2.)G > ,
3*64,:GA6 < U
```

By the way, if you try this, the commas and the PET INPUT statement will hang you up. Replace line 50 with:

```
50 PRINT: PRINT"dn? sp sp";
   : M$ = ""
51 GET A$: IF A$ = "" THEN 51
52 IF A$ = CHR$(13) THEN 60
53 M$ = M$ + A$; PRINT A$; :
   GOTO 51
```

If you wish, add a false cursor too. This version unfortunately lacks any kind of editing if you make a mistake...

The next thing is to make the code harder to break - this code is very easy to break. One way is to provide a keyword which is used to compute the amount to shift the alphabet for each character. For example, CAT would make S = 3 for the first letter, S = 1 for the second letter, etc. When the keyword is exhausted, it is repeated (i.e., CATCATCATCAT).

The next program uses the keyword method and adds some frills. You can enter a fairly long message which is stored in a string array and then the PET will display the encrypted version of the entire message. The code is broken into 5 letter blocks for easier transcription (a standard method that spies used to use).

When you RUN the program in Figure 1, the choice of Encode or Decode is presented and then the keyword is requested. See if you can follow the sequence below:

```
RUN
(screen clears)
BETTER PET CIPHER PRO-
  GRAM
CIPHERKEY? AGENT 9
(screen clears)
ENTER YOUR MESSAGE TO
  ENCODE
'EXIT' WHEN DONE
/ IGOR COMMANDS YOU TO
  COME NOW!
/ EXIT
(screen clears)
CODE GROUPS:
JNT 4 #HNTF \X3YZ VZ. (/
  YDVRS 4.HX(
PRESS ANY KEY
(Press STOP to leave this pro-
  gram)
```

Now to decode the message - RUN again and choose the DECODE option and the same keyword, AGENT 9:

```
ENTER YOUR CODE GROUPS
WITH A SPACE BETWEEN
EACH GROUP. 'EXIT' TO STOP.
/ JNT 4 #HTNF /X3YZ VZ. (/
/ YDVRS 4.HX(
/ EXIT
(screen clears)
YOUR MESSAGE:
IGOR COMMANDS YOU TO
COME NOW!
PRESS ANY KEY
```

If you play with this, you will soon realize that correct entry of coded information is a formidable task. It took me 5 tries to get the example shown above correct.

The improved cipher program is a **lot** longer. Most of its bulk comes from attempting to make the program easy to use and fairly tolerant of errors. Here is a detailed look at how the program works. If you are a beginner, you might want to skip this, or at least skim the difficult parts.

The array M$ and C$ are to hold the messages and the code strings, respectively. Lines 10 to 90 ask for the choice between encoding and decoding and then request the cipherkey string K$. Line 75 sets two values for use by the encoding and decoding subroutines in 2000 and 2500. KP (keypointer) scans along the keystring for the character used for encode/decode, and KL (Keylimit) is

Pet, con't...

the number of characters in the keystring.

Lines 500 to 760 take care of the encoding part of this program.

Lines 500 to 540 enter your message into the array M$. Note that a special input routine, 3000, is called. Line 550 is a relic from a previous version of this program and can probably be discarded with no ill effects. (But why kick a working program? It might bite!)

Line 560 serves a special purpose. Later, the array M$ will be scanned and encoded. Either exiting the loop or entering 'EXIT' leaves J, the number of strings in M$ too large by one. (I had some trouble catching this one.)

Lines 570 to 650 take each M$ string and converts it into an equivalent code string C$. Since the sequence of entering some message and coding it is repeated, Lines 520 and 570 ensure that M$ and C$ are cleared of any previous "garbage" before use. Subroutine 2000 converts the character C$ into the encoded character C$ (i.e., after 2000, C$, is different).

To make neat groups of 5 characters, the coded string in the array C$ must be a multiple of 5 characters. Lines 620 to 640 check for this and add encoded blanks as needed.

When all that is done, the array C$ holds the encoded message. Lines 650 to 730 then display the groups on the PET screen. The C$ strings are broken up into 5 character blocks and put on the screen. CG counts the number of 5 character blocks in one string from C$(). CP computes where to pick out the block with MID$ (lines 690 and 700) and DG keeps track of how many groups have been printed to the screen. (40/6 = 6 groups per PET line - remember a space has to separate the groups.)

Lines 740 to 750 wait for your keypress and then line 760 starts things all over. (If you wish, test for A$ = "X" and jump to line 30 for a nicer version.)

The reason for all the code in 500 to 760 is to make the program easier to use. Several lines of message are encoded at once and the code is output in neat, 5 character groups for easy transcription. However, the price for all this niceness is a bit high...

If you understand 500-760, Lines 1000 to 1160 read as a simplified version and here the conversion is from C$() to M$(). Lines 1020 to 1050 enter the code groups into C$ - again with the input routine 3000. Lines 1070 to 1110 decode the array C$() into M$() (note that line 1080 skips every 6th character which is the space used to separate code groups). Subroutine 2500 is used for decoding the letter C$.

Routine 2000 checks C$ (the PET is **very** nasty on string functions applied to null strings), moves the keypointer KP up by one (if KP runs off the edge of the string, it is set to 1 to start over).

Line 2020 is the actual encoding step. Here the letter in K$ is used to choose how far to "slide" the alphabet for C$. Lines 2030 and 2040 take care of "wrap-around." If Z were encoded with Z, the result is too large, and the subtraction makes the new value a character which can be entered from the keyboard. Routine 2500 does the same operations, with Line 2520 decoding instead.

Routine 3000 is a mess. This permits you to input a string without regard to quotes, colons, or commas (which INPUT likes to do funny things with) and allows the editing character DEL to remove the last letter entered.

You might want to use this routine in other programs, so enter and debug it first - and save on a tape for other uses. Line 3000 prints a prompt (the slash which is above the I key), clears B$, and sets Q$ to the quote mark (").

Lines 3010 to 3040 provide a "false cursor" (which has been described in a previous column). Once a non-null character has been found, it is checked for RETURN (Line 3100). A RETURN produces a PRINT to echo the effect of pressing RETURN and the routine ends. (Don't forget this one - I spent an hour wondering why RETURN didn't work and found out it was working all the time, but I didn't **see** it on the screen.) Lines 3100 to 3150 check for DEL, and depending on how many characters are in B$, removes the last character and prints the DEL on the screen. Line 3120 prevents the DEL being echoed for empty B$. (Why delete from an empty string and keep the 'cursor' from moving up a line?) Line 3130 is needed for the PET's inability to produce null strings via string functions. Line 3140 does the real work here (after all the maddening exceptions).

Line 3160 is a quaint way to echo a quote mark. Only the DEL character will backspace after the PET is in quote mode, so the quote is printed and then deleted. Line 3170 removes graphics characters and the various other screen editing functions by refusal to enter into B$ or echoing. Line 3180 echoes A$ (your keystroke), and enters it into B$. Note that if A$ is a quote, it is printed (a second time) and the PET is now **OUT** of quote mode (neat!).

Space runneth out, and I will do more on cryptic stuff later. The next column will cover some more on BASIC - this time the variables and how they are stored.

Correspondence is welcome. Please include a SASE and mail to Gregory Yob, Personal Electronic Transactions, Box 354, Palo Alto, CA 94301. All letters will be acknowledged. □

POWERSOFT, INC.

P. O. BOX 157
PITMAN, NEW JERSEY 08071
(609) 589-5500

products for the
APPLE II

APPLESOFT II UTILITY (Diskette Only) $12.45

The Applesoft II Utility program provides the user with the following features. a) Complete automatic renumbering of any Applesoft II program. b) The creation of an EXEC File for subroutine file creation. This feature allows you to incorporate the same subroutine in various programs. c) No modification of the program in machine memory (RAM). d) Automatic running of the program. No programmer should be without this excellent utility program. REQUIREMENTS: Disk II, Applesoft II, 16K of memory.

REAL ESTATE ANALYSIS PROGRAM $14.95

The Real Estate Analysis Program provides the user with three features. a) A powerful real estate investment analysis for buy/sell decisions and time to hold decisions for optimal rental/commercial investments. b) Generation of complete amorization schedules consistent with banking practices and schedules. c) Generation of depreciation schedules for selecting the best depreciation schedule for your use and a determination of optimal switch over points to straight line to maximize depreciation. All three features are designed for video screen or printer output. In addition, the program will plot; cash flow before taxes vs. years, cash flow after taxes vs. years, adjusted basis vs. years, capital gains vs. years, pre-tax proceeds vs. years, post-tax proceeds vs. years, and return on investment (%) vs. years. REQUIREMENTS: Applesoft II, 16K of memory without DOS or 32K of memory with DOS (Disk II).

ADDRESS FILE GENERATOR $19.95

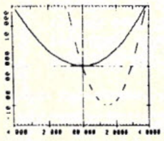
A professional piece of software which allows the user to create four different types of address files: a) Holiday File, b) Birthday File, c) Home Address File, and d) Commercial Address File. The program contains a menu of seven major commands: 1) Create a File, 2) Add to File, 3) Edit File, 4) Display File, 5) Search File, 6) Sort File, and 7) Reorganize File. Most of the major commands have subordinate commands which adds to the flexibility of this powerful software system. We doubt you could buy a better program for maintaining and printing address files. REQUIREMENTS: Disk II, Apple Printer Card, 32K of memory with Applesoft ROM Card or 48K of memory without Applesoft ROM Card.

SUPER CHECKBOOK $19.95

A totally new checkbook program with a unique option . . . Bar Graphs. These bar graphs, outputed to a printer or video screen, provide trend analysis data on code expense, income, expenses, or gain/loss on a month by month basis. The program contains a total of fourteen options: 1) Check/Deposit Entry & Modification, 2) Reconciliation of Checks or Deposits, 3) Sort by Check Number, 4) Sort by Code for Year, 5) Sort by Code for Month, 6) Output Year to Date, 7) Output Month Activity, 8-11) Printer/Video Plot Trend Analysis-Bar Graphs, 12) Account Status, 13) Reconciled Check Status, and 14) Quit. An excellent program for maintaining your checkbook, or that of a small business. REQUIREMENTS: Disk II, 32K of memory with Applesoft ROM Card or 48K of memory without Applesoft ROM Card.

FUNCTION GRAPHS AND TRANSFORMATIONS $14.95

This program uses the Apple II high resolution graphics capabilities to draw detailed graphs of mathematical functions which the user defines in Basic syntax. The graphs appear in a large rectangle whose edges are X and Y scales (with values labeled by up to 6 digits). Graphs can be superimposed, erased, drawn as dashed (rather than solid) curves, and transformed. The transformations available are reflection about an axis, stretching or compressing (change of scale), and sliding (translation). The user can alternate between the graphic display and a text display which lists the available commands and the more recent interactions between user and program. Expected users are engineers, mathematicians, and researchers in the natural and social sciences; in addition, teachers and students can use the program to approach topics in (for example) algebra, trigonometry, and analytic geometry in a visual, intuitive, and experimental way which complements the traditional, primarily symbolic orientation. REQUIREMENTS: 16K of memory with Applesoft ROM Card or 32K of memory without Applesoft ROM Card.



Available at your local computer store

Call or write for our free SOFTWARE & ACCESSORIES CATALOG

DEALER INQUIRIES INVITED

POWERSOFT, INC.
P. O. BOX 157
PITMAN, NEW JERSEY 08071
(609) 589-5500

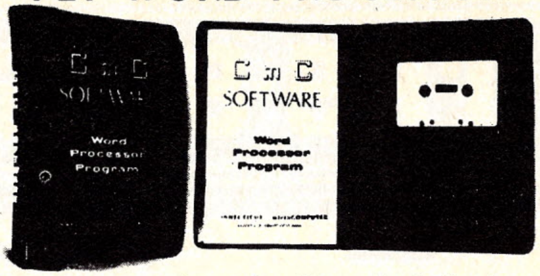
- Check or Money Order
- Include $1.00 for shipping and handling
- C.O.D. ($1.00 add'tl. charge)
- Master Charge and VISA orders accepted
- New Jersey residents add 5% sales tax

Programs Available on Diskette at $5.00 Additional

Apple II is a registered trademark of Apple Computer, Inc.

CIRCLE 229 ON READER SERVICE CARD

PET WORD PROCESSOR



This program permits composing and printing letters, flyers, advertisements, manuscripts, etc., using the COMMODORE PET and a printer.
Script directives include line length, left margin, centering, and skip. Edit commands allow the user to insert lines, delete lines, move lines and paragraphs, change strings, save onto cassette, load from cassette, move up, move down, print and type.
The CmC Word Processor Program addresses an RS-232 printer through a CmC printer adapter.
The CmC Word Processor program is available for $29.50. Add $1.00 for postage and handling per order.

Order direct or contact your local computer store.

VISA
master charge

CmC
CONNECTICUT microCOMPUTER
150 POCONO ROAD
BROOKFIELD, CONNECTICUT 06804
(203) 775-9659

CIRCLE 132 ON READER SERVICE CARD

TRS-80® BUSINESS SOFTWARE

TRS-80 is a registered trademark of Radio Shack

OSBORNE & ASSOCIATES ADAPTATIONS

Complete Business System

- ACCOUNTS PAYABLE $75.*
- ACCOUNTS RECEIVABLE $75.*
- GENERAL LEDGER $75.*

Continuous Forms and Checks Available

A SMALL BUSINESS SYSTEM CAN MAKE A BIG DIFFERENCE

All Prices Subject to Change without Notice.

PAYSOFT™ PAYROLL SYSTEM

- FIELD PROVEN - Systems throughout the U.S.
- SIMPLIFIED OPERATOR INTERFACE
  No training required
  Fast - minimal system operation time
- AUTOMATED REPORTS
  Check writer/complete paystub
  Employee summary - company summary
  Check register
  Federal deposit record
  Quarterly & annual federal forms (optional)
- EDITING FEATURES
- BONUS PAY ENTRY FEATURE
- END-USER DOCUMENTATION - End-user support
- PAYSOFT DISKETTE AND MANUAL $125.*
  ANNUAL REVISION SUBSCRIPTION OPTIONAL

BUSINESS MICROPRODUCTS
A DIVISION OF THE READY CORPORATION
609 S. Livermore Ave • Livermore, Ca 94550

*California residents add 6.5% sales tax

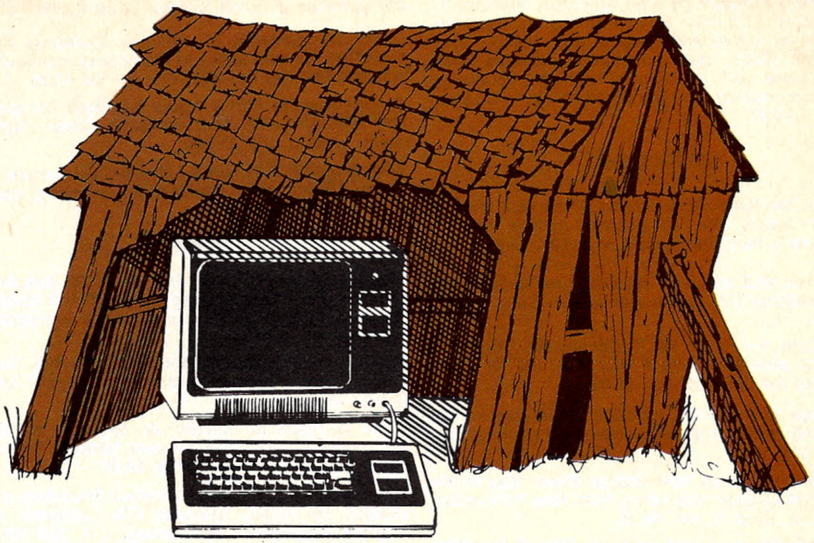
BAC/VISA/MC accepted

Sample Reports Send $2.00
Add $3.00 per Order for Shipping and Handling

CIRCLE 115 ON READER SERVICE CARD

TRS-80 Strings

Stephen B. Gray



For column number 10, we get a little further into TRS-80 graphics with a kaleidoscope program or two, then take a look at the fascinating Electric Paintbrush, and end with a challenge to hotshot TRS-80 programmers.

Kaleidoscope Graphics

When we first got into graphics in this column, we described a short program that randomly filled the screen with graphics blocks, using the full display area. Then we went on to make it smaller, and to duplicate the same pattern in an allover design, like some fabric or wallpaper designs.

Now it's time to look at what can be done with a small pattern in a kaleidoscope type of design. First we write a program which will create a small pattern that changes continually:

```
100 CLS
110 X = RND(20)
120 Y = RND(10)
130 SET(X,Y)
140 RESET(X + 1, Y + 1)
300 GOTO 110
```

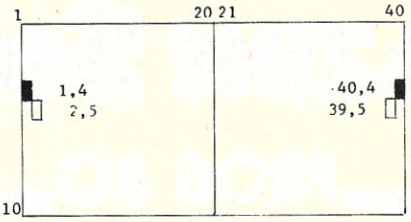
which puts a never-the-same-twice pattern in the upper left corner of the TRS-80 screen.

Let's create a mirror image of this pattern, just to the right of it. That means we want to, in effect, flip the pattern over, like a page in a book, to the right. The Y-coordinate will be the same, but the X-coordinate will be a mirror image.

(In the following discussion, for the sake of simplicity we'll assume that the first usable column at the left of the screen in the display area is the one-column, instead of the zero-column.)

What is the left-to-right mirror image of a graphics block that's at position 1,4? If it's 4 down from the top, then the mirror-image block will also be 4 down from the top, for the Y-coordinate. If its X-coordinate is 1, then it's in the first column at the left edge, and its left-to-right mirror image will be in the column at the right edge.

Because the basic pattern we created is 20 blocks wide, the right edge of the mirror image will be at the 40th position, so the left-to-right mirror image of 1,4 is 40,4.



If we want the two patterns to be right up against each other, then we must use

```
150 SET(41-X,Y)
```

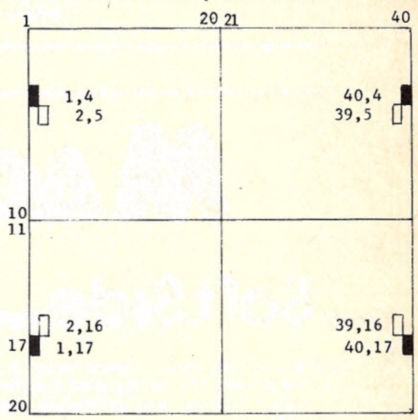
to create the left-to-right mirror image. If the block to be reset by line 140 is at the southeast of block 1,4 then the block to be reset in line 160 is at the southwest of block 40,4 or at 39,5 so we have

```
160 RESET(40-X,Y + 1)
```

Add these to the program, and RUN to see if it works. It should produce a plesant design, but one that somehow looks half-finished. So let's finish it.

We can add a top-to-bottom mirror image to this two-way pattern, to make it a four-way pattern, by simply adding two more pairs of SET/RESET lines.

The image at the bottom left will have the same X-coordinate as the image at the top left. If the block in the top-left image is at the fourth position down from the top, then the corresponding top-to-bottom mirror-image block in the bottom-left image is four positions up from the bottom. That bottom edge is at 20, since the basic pattern is 10 blocks high. And the block that's four positions up from



position 20 is at position 17, or at X-Y position 1,17. The SET line is similar to the SET line for the top-right image:

```
170 SET(X,21-Y)
```

but it creates a top-to-bottom mirror image. Now the block to be reset is at the northeast of block 1,17, or at 2,16 so we have

```
180 RESET(X + 1, 20-Y)
```

Now add this pair to the program, and RUN.

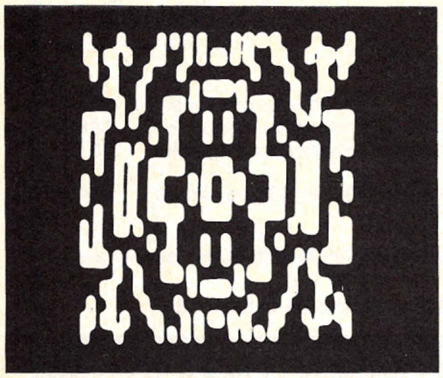
As for the bottom-right image, we've already calculated the position for the blocks to be set and reset there, since their X-coordinates are the same as for the top-right image, and their Y-coordinates are the same as for the bottom-left image:

TRS-80 Strings, con't...

```
190 SET(41-X,21-Y)
200 RESET(40-X,20-Y)
```

which puts the graphics block in this quadrant at 40,17 and resets the block that's at 39,16.

Now you've got the whole four-way, quadrilateral kaleidoscope, and if you run all four pairs of SET/RESET lines, you'll have a pleasing pattern.



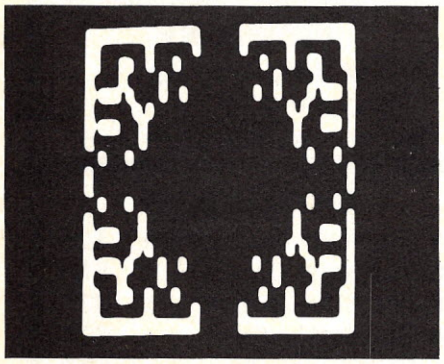
This design is not quite square, because the aspect ratio of the graphics block is close to 3:7, so if you really prefer a square design, you should use 21,9 instead of 20,10 as the size of the basic pattern.

In the kaleidoscope program, the resulting design has graphics blocks at totally random locations. To skew the pattern blocks toward the four corners of the design, change lines 110 and 120 to:

```
110 X = RND(RND(20))
120 Y = RND(RND(10))
```

and even more toward the corners with

```
110 X = RND(RND(RND(20)))
120 Y = RND(RND(RND(10)))
```



and then figure out how this random-random effect works.

The basic pattern can be altered somewhat by changing lines 110 and 120 to

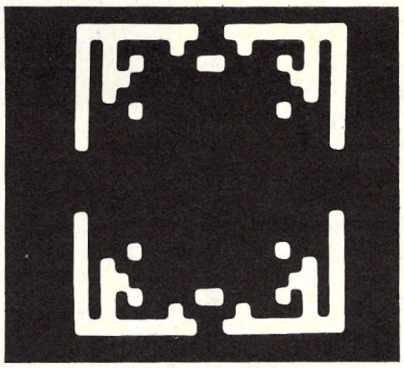
```
110 X = RND(INT(SQR(400)))
120 Y = RND(INT(SQR(100)))
```

and there are many more such combinations that can be used to create variations on the basic theme.

If you add

```
105 PRINT CHR$(23)
```

you'll get a design made up of double-width graphics blocks. But if the number in line 110 is odd, the design will no longer be in quadrilateral sym-



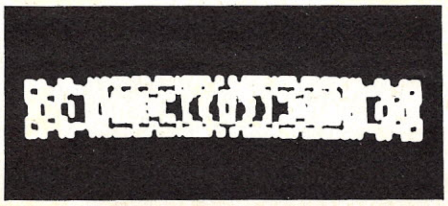
metry. It will be only bilateral, with top-to-bottom symmetry. Can you figure out why?

As in the program for the allover design, this kaleidoscope design can be made "emptier" or sparser, by adding more RESET lines to put more black into the pattern. But they have to be added in groups of four, for the four-way symmetry.

A general kaleidoscope program is easily written, so that designs of any size can be created.

```
100 CLS
110 INPUT A,B
120 CLS
130 X = RND(A)
140 Y = RND(B)
150 SET(X,Y)
160 SET(2*A-X+1,Y)
170 SET(X,2*B-Y+1)
180 SET(2*A-X+1,2*B-Y+1)
190 RESET(X+1,Y+1)
200 RESET(2*A-X,Y+1)
210 RESET(X+1,2*B-Y)
220 RESET(2*A-X,2*B-Y)
230 GOTO 130
```

This program permits making a four-way kaleidoscope design of any size, although square ones, for example, with A,B values below 14,6 or so are too small to be interesting, and those with A,B values above 28,12 or so are too big for the eye to take them in all at once and be pleasing. For openers, try A,B values of 60,4 for a skinny horizontal design,



7,23 for a skinny vertical one, 63,23 to see why big is not always beautiful, and 20,10 for a pleasing effect.

Incidentally, the program runs faster if you use A + A instead of 2*A. The general kaleidoscope program could have included a few lines that would automatically center the design.

To center the design, first add 63 to the X-coordinate and 23 to the Y-coordinate, to move the top-left corner of the design to the center of the screen. Then subtract A from the X value and B from the Y value, to adjust the design's position, by moving it halfway left and halfway up, so that, no matter what its size, it's always centered on the screen. Lines 150 and 190, when changed in this manner, are like this:

```
150 SET(X+63-A,Y+23-B)
190 RESET(X+64-A,Y+24-B)
```

Can you change the other three pairs of SET/RESET lines in the same way?

Electric Paintbrush

This real-time graphics language is from Personal Software (592 Weddell Drive, Sunnyvale, CA 94086), the people who produced the fantastic Microchess 1.5 (Feb. 1979, p. 102), and who are asking only $14.95 for a program whose capabilities will dazzle you.

Written by Ken Anderson, Electric Paintbrush operates in 4K of RAM on both Level-I and Level-II TRS-80 machines. It's written in machine language, which means the graphics are drawn far faster than possible with BASIC.

The commands are few but powerful, to let you draw lines, turn corners, change white to black and vice versa, repeat previous steps, or call other programs. You can group several commands into a single new command, only a few of which are needed to draw amazingly complex patterns.

For instance, press the F key, and a graphics block appears in the center of the screen. Each time you press F, which moves the brush forward one stroke, you add one more block to the line, heading north. The line will run off the top of the screen, if you press F enough times, then wrap around, and come up from the bottom of the screen. Press R, which rotates the direction of the brush clockwise 45 degrees, and the next time you press F, the line will head in a northeast direction. If you'd pressed R twice,

apple tv & computing

2606 South Robertson Blvd.
LOS ANGELES, CALIFORNIA 90034
(213) 559-4268

PET MEANS BUSINESS!!

The new COMMODORE CBM with Intelligent Dual Minifloppy and Tractor impact printer is taking the lead in its price class!

CBM WORD PROCESSOR

is a FULL FEATURE Direct Cursor Editor, with features like auto repeat, auto insert and delete, global search and find, scrolling, full justification.

It's superfast formatter and output section gives you capabilities of tremendous throughput at lower cost!

Couple one to our Centronics 753 printer and you have 130 to 150 CPS!!

Also available for the CBM:
GEN'L LEDGER, A/P-A/R, PAYROLL
Based on Osborne & Assoc. books, these are well documented, efficient programs. $125 per Module, or... only $350 complete pack.

MasterCharge, Visa Welcome

CIRCLE 110 ON READER SERVICE CARD

PET™: CURSOR Programs for PET™ Computers

The Original Cassette Magazine for the Commodore **PET.**

- Five excellent programs come to you each month on a C-30 cassette, ready to load and run.
- Every issue has an animated "Front Cover" that uses **PET** graphics to the fullest.
- Compare value: each issue of CURSOR includes a featured program that would cost $7.95 to $12.95 elsewhere!
- CURSOR is mainly entertainment, but we also publish useful utility programs, as well as educational and business programs.
- With each monthly CURSOR cassette, you also receive CURSOR notes with written instructions for the programs, and a fresh, opinionated look at our crazy industry.

CURSOR = High Quality
PET Software

☐ 12 issues only $33 in US & Canada
☐ 6 issues for $20
☐ Sample Copy $3.95
We Accept VISA and Mastercharge

Name ____________________

Address ____________________

City ____________________

State/Zip ____________________

CURSOR, Box 550, Goleta, CA 93017

Our Subscribers are Happy People!

CIRCLE 136 ON READER SERVICE CARD

EDUCATIONAL SOFTWARE
TRS80 & Pet

. Elementary
. Math
. Business
. Accounting
. Social Studies
. Economics
. Biology
. Games

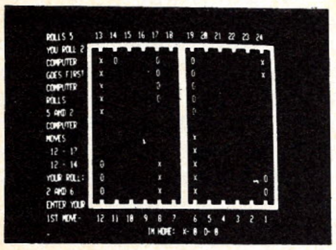
Write for catalog
Micro Learningware
Box 2134
N. Mankato, MN 56001

CIRCLE 164 ON READER SERVICE CARD

Improve Your Game With

>>> FASTGAMMON >>>®

An Exciting New Backgammon Opponent!



>>> FASTGAMMON >>> ON THE TRS-80

Ready to run on your

- **TRS-80** (level II)
- **APPLE II** (16K)
- **SOL**
- **POLY-88**
- **COMPAL-80**

All versions require at least 16K. Apple disk requires 24K. Sol disk North Star only. Compal disk Micropolis dual density only. Poly-88 not available on disk.

OUTSTANDING FEATURES! — Computer makes good moves instantaneously. Literal and graphic displays of each move. Option to replay same rolls. Eight-page instruction manual.

OUTSTANDING VALUES! — Only $20 on cassette or $25 on diskette. California residents add 6% sales tax.

SEE IT NOW AT YOUR LOCAL COMPUTER DEALER OR ORDER DIRECT FROM

QS Quality Software

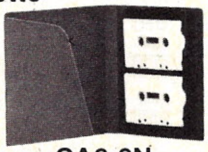
10051 Odessa Avenue
Sepulveda, CA. 91343
Dealers: For info call (213) 344-6599

CIRCLE 184 ON READER SERVICE CARD

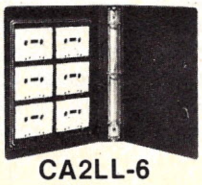
RELIANCE

America's Most Versatile Line of Customized Audio Visual Packaging

Audio Cassette Albums



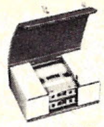
CA2-2N



CA2LL-6

Corrugated Mailers

M-1



M-2

For complete catalog write or phone:

RELIANCE Plastics & Packaging Division
108-18 Queens Boulevard
Forest Hills, N.Y. 11375 (212) 544-9800

CIRCLE 190 ON READER SERVICE CARD

TRS-80* APPLE II OWNERS**

"We Can Make You Smarter"

With the **INDIVIDUAL STUDY CENTER** you can teach yourself any subject you wish. Whether you are 6 or 60 we have tapes on subjects you want to learn. With the **INDIVIDUAL STUDY CENTER** you simply load the tape in and you can use it with a variety of activities. Do you like crossword puzzles? Then use **PUZZLER** to learn French or Spanish. Do you have a child who needs to learn spelling or do math exercises? Why not make studying fun? Use **HOUSE ON FIRE** and let your child study using your computer. Our exciting activities have interesting graphics, involve the learner, and reinforce the right answer. Does your teen-ager hate studying? Let him try **AROUND THE BALL PARK** with history facts.

The **INDIVIDUAL STUDY CENTER** contains 2 computer cassettes in a sturdy vinyl binder, and a manual with simple instructions and descriptions of every activity. **BONUS** -we include a blank cassette, a sample data tape and a special **MAINTENANCE** program that allows you or your child to prepare your own information subject tapes. You only need to make a tape once and you can study the information with many different activities including the traditional matching and completion. **INDIVIDUAL STUDY CENTER** only $39.95 plus $1.50 postage and handling Tapes Guaranteed. **TYC SOFTWARE** tm has many prepared subject tapes. More being added daily.

STILL AVAILABLE-MIND-Step 1 and 2-MEMORY IMPROVEMENT COURSE. Using graphics and good psychological techniques you can learn to improve your ability to remember lists of facts, or more detailed information. Also included are techniques for remembering names and faces and listening skills.

SEND FOR **free** BROCHURE TODAY!

TYC SOFTWARE tm
40 Stuyvesant Manor Geneseo, New York 14454

*A trademark of Tandy Corp.
**A trademark of Apple Computer Inc.

VISA

CIRCLE 201 ON READER SERVICE CARD

TRS-80 Strings, con't...

the line would head east next time you pressed F.

You can do a great deal with just F and R, which are the two most important brush-movement commands, but the D command begins to automate the process by letting you define your own drawing commands.

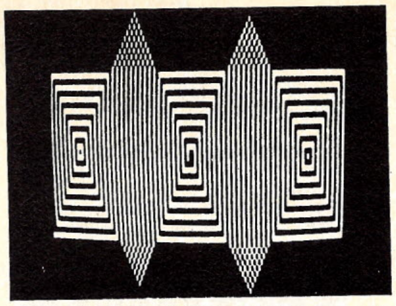
For example, type D#(10F2R), and now every time you type #, the paintbrush will follow the newly defined sequence of 10 forward steps followed by two rotates. So if you type # four times, you've got a rectangle. (You'd get a square if the graphics blocks were themselves square, but because they have an aspect ratio of 3:7, you get a rectangle that's a little more than twice as high as it is wide.)

The well-written manual notes that "if you continue to press #, you will make the original rectangle disappear. This is because you are in the S, or switch trace, mode. When painting on a white screen, a black brush stroke is left, and when painting on black, a white line is deposited. W and B are two other modes which may be entered to leave a white or a black trace on the screen."

With two user-defined commands, you can paint an ever-increasing spiral. First, type DG (SHNCA-) and you've defined G as: set the switch mode (S), move the paint brush to the center of the screen (H), head the brush north (N), clear the screen (C) and decrement the accumulator A times (this sets the accumulator to zero).

Second, type DX(+2RAF) and you've defined X as: increment the accumulator once (+), rotate the brush twice for a 90-degree turn (2R), and move forward A steps (AF). Now each time you press X, the brush will paint one more line in an in ever-increasing spiral.

To automate the spiral-painting process fully, type D!(G1000X), which defines ! as the draw-a-spiral command. Now just type ! and the G command will be repeated 1000 times. "When the paintbrush crosses the edge of the screen, it reappears at the opposite edge," according to the manual. "As it crosses old white lines, it paints them black. The spiral keeps folding over itself to create an exciting op-art effect." Which means that once the spiral has filled the screen, the paintbrush keeps right on going, and turns the original spiral into a continuing variety of patterns that are quite fascinating as well as unpredictable.



Other commands enlarge your pallette of pattern possibilities, such as the random command ? that executes one of two commands on a random basis, pen down and pen up (P and U) which either leave a trace or no trace, S to save the current user-defined commands on cassette, L to load new commands from cassette, and several more.

The manual gives sets of commands for drawing a stylized dragon, a Hilbert curve, and "windmills." From there on, you're on your own, and if you have any artisitic or math talent at all, you should be able to come up with some graphics that will amaze your friends, as well as keep you out of mischief for many hours.

As far as I'm concerned, there's only one command missing: one that would let you freeze the pattern so you can look at this or that particular design awhile. If you try to use SHIFT in Level-II, you'll stop the pattern all right, but you can't make it continue from that point; you have to start it going from the beginning again.

The only way to make a continuously-changing pattern stop at a certain point is to figure out exactly how many repeats of the basic element are needed to arrive at that point, and then change the number of repeats in the appropriate user-defined command so that the pattern stops there.

Look for Electric Paintbrush at your local computer store, and if they don't have it, send $14.95 to Sunnyvale. If you thought Personal Software was in Cambridge, MA, it was; until July 1 (officially), that is; then it moved to CA.

Programmer Revealed

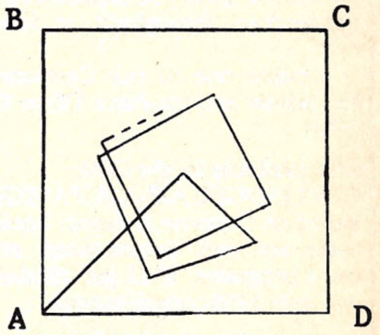
The two short programs on page 126 of the June 1979 issue (sinewaves and "backwards") were received without a signature on the accompanying letter. The author has now written to say he's Jim Raden of Maumee, Ohio, and that he's just turned 14.

Software Challenge #1 —Square Within a Square

Think you're a pretty good BASIC programmer? Hot on TRS-80 graphics? Here's a challenge. It's not a contest; there are no prizes, other than the satisfaction of writing a program that leads the TRS-80 through a complex task. Like virtue (or vice), the program is its own reward.

Put a square on the screen, then start running a line from any corner to the opposite corner. But stop halfway across, and then aim at the next corner, clockwise. Again, stop halfway there, and aim at the third corner, clockwise.

That is, start at A, go halfway toward C, then halfway toward D, then halfway toward A, then halfway toward B, etc.



Although the first few lines seem to have no relationship to each other, quite soon the lines begin to trace a square, and from then on will just retrace the square over and over. In theory, that is. But will the lines result in an eternal square in fact?

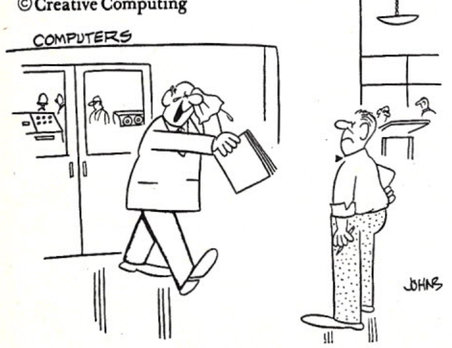
Writing a program to draw the square ABCD is easy. But can you carry on from there, and draw the "halfway lines?" As an added touch, can you clean up the display by making the early halfway-lines disappear, or even better, fade away? This would eventually result in a display of just the smaller square inside the larger ABCD square.

And for you geometers, can you determine the relationship between the length of a side of the smaller square, and side AB of the larger square? Do it with a pencil and paper first, then check your findings be measuring the lines on the screen.

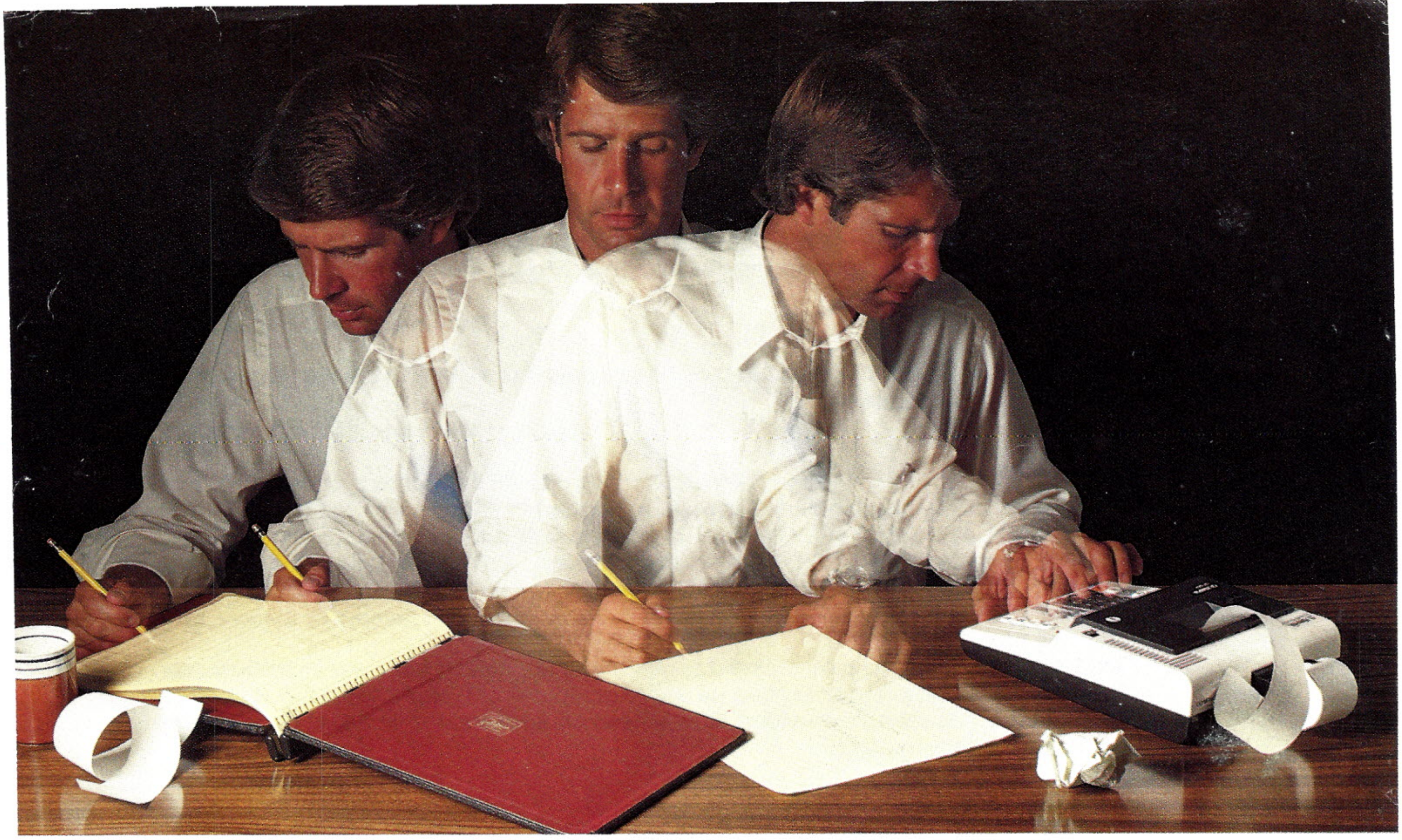
If you can write this program, and would like to see it printed in this column, please send me a printout of the program, dark enough to be printed on these pages, heavy with REMs or with accompanying documentation, and a short cassette of the program.

More software challenges will be forthcoming, challenges that make this one look simple. Which it will be, if you're half the programmer your friends say you are... □

Index To Advertisers



"Wait'll I get control of myself and I'll tell you what the computer thought of your design specifications!"



Solve your personal energy crisis. Let VisiCalc™ Power do the work.

With a calculator, pencil and paper you can spend hours planning, projecting, writing, estimating, calculating, revising, erasing and recalculating as you work toward a decision.

Or with VisiCalc and your Apple* II you can explore many more options with a fraction of the time and effort you've spent before.

VisiCalc is a new breed of problem-solving software. Unlike prepackaged software that forces you into a computerized straight jacket, VisiCalc adapts itself to any numerical problem you have. You enter numbers, alphabetic titles and formulas on your keyboard. VisiCalc organizes and displays this information on the screen. You don't have to spend your time programming.

Your energy is better spent using the results than getting them.

Say you're a business manager and want to project your annual sales. Using the calculator, pencil and paper method, you'd lay out 12 months across a sheet and fill in lines and columns of figures on products, outlets, salespeople, etc. You'd calculate by hand the subtotals and summary figures. Then you'd start revising, erasing and recalculating. With VisiCalc, you simply fill in the same figures on an electronic "sheet of paper" and let the computer do the work.

Once your first projection is complete, you're ready to use VisiCalc's unique, powerful recalculation feature. It lets you ask "What if?", examining new options and planning for contingencies. "What if" sales drop 20 percent in March? Just type in the sales figure. VisiCalc instantly updates all other figures affected by March sales.

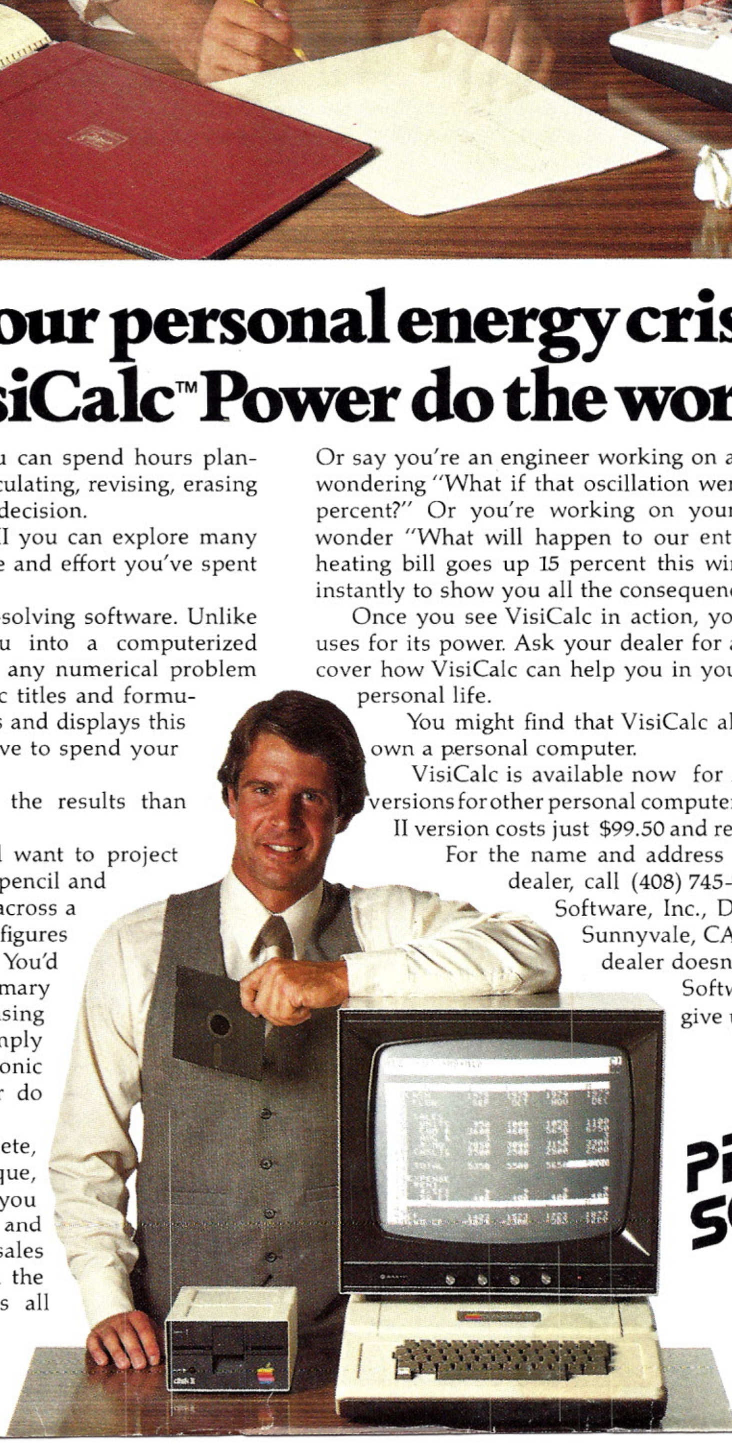
Or say you're an engineer working on a design problem and are wondering "What if that oscillation were damped by another 10 percent?" Or you're working on your family's expenses and wonder "What will happen to our entertainment budget if the heating bill goes up 15 percent this winter?" VisiCalc responds instantly to show you all the consequences of any change.

Once you see VisiCalc in action, you'll think of many more uses for its power. Ask your dealer for a demonstration and discover how VisiCalc can help you in your professional work and personal life.

You might find that VisiCalc alone is reason enough to own a personal computer.

VisiCalc is available now for Apple II computers, with versions for other personal computers coming soon. The Apple II version costs just $99.50 and requires a 32k disk system.

For the name and address of your nearest VisiCalc dealer, call (408) 745-7841 or write to Personal Software, Inc., Dept. C, 592 Weddell Dr., Sunnyvale, CA 94086. If your favorite dealer doesn't already carry Personal Software products, ask him to give us a call.



PERSONAL SOFTWARE

TM—VisiCalc is a trademark of Personal Software, Inc.
*Apple is a registered trademark of Apple Computer, Inc.

CIRCLE 178 ON READER SERVICE CARD

Don't take our word for it.

"We can heartily recommend the Superboard II computer system for the beginner who wants to get into microcomputers with a minimum of cost. Moreover, this is a 'real' computer with full expandability."

Popular Electronics March, 1979

"(Their) new Challenger 1P weighs in at $349 and provides a remarkable amount of computing for this incredible price."

Kilobaud Microcomputing February, 1979

"Over the past four years we have taken delivery on over 25 computer systems. Only two have worked totally glitch free and without adjustment as they came out of the carton: The Tektronic 4051 (at $7,000 the most expensive computer we tested) and the Ohio Scientific Superboard II (at $279 the least expensive) . . . The Superboard II and companion C1P deserve your serious consideration."

Creative Computing January, 1979

"The Superboard II and its fully dressed companion the Challenger 1P series incorporate all the fundamental necessities of a personal computer at a very attractive price. With the expansion capabilities provided, this series becomes a very formidable competitor in the home computer area."

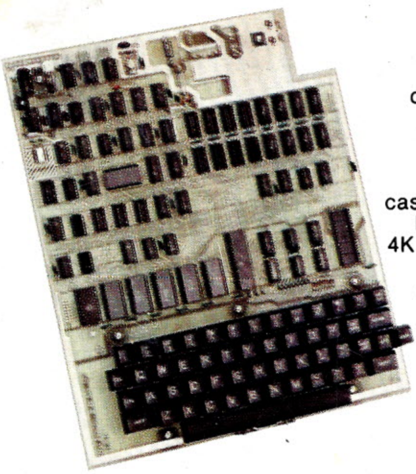
Interface Age April, 1979

"The graphics available permit some really dramatic effects and are relatively simple to program . . . The fact that the system can be easily expanded to include a floppy means that while you are starting out with a low-cost minimal system, you don't have to throw it away when you are ready to go on to more complex computer functions. Everything is there that you need; you simply build on to what you already have. You don't have to worry about trading off existing equipment to get the system that will really do what you want it to do. At $279, Superboard II is a tough act to follow."

Radio Electronics June, 1979

"The Superboard II is an excellent choice for the personal computer enthusiast on a budget."

Byte May, 1979



SUPERBOARD II
$279.00
The world's first complete computer system on a board including full keyboard, video display, audio cassette interface, 8K BASIC-in-ROM and 4K RAM. Expandable. Requires +5V at 3 amp power supply.

C1P $349.00
Complete with enclosure and power supply. All features of Superboard II. Easy to expand to more memory and floppy disk.

C1P MF $995.00
The first floppy disk based computer system the world has ever seen for under $1,000. 8K BASIC-in-ROM, 12K RAM. Expandable to 32K RAM.

OHIO SCIENTIFIC

1333 S. CHILLICOTHE RD., AURORA, OHIO 44202 (216) 562-3101

See your Ohio Scientific dealer for full details.

CIRCLE 171 ON READER SERVICE CARD